HUMAN OPTIMAL CONTROL FOR AIRCRAFT FLYING VARIOUS STEADY STATE MANEUVERS

By

JUERG MUELLER

Dipl. Masch.-Ing. ETH

Swiss Institute of Technology

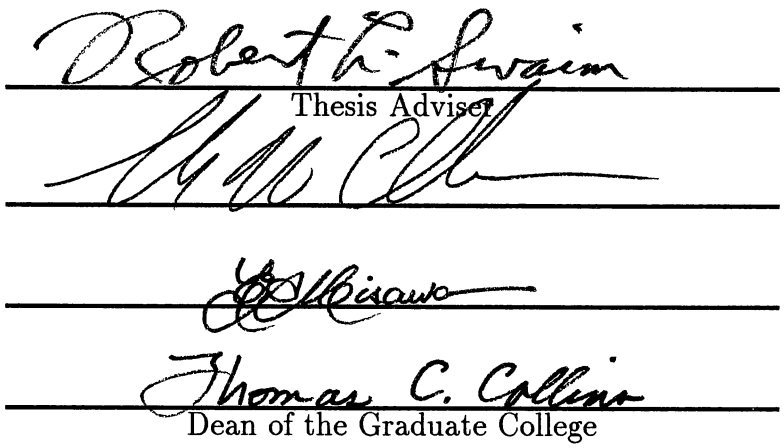
Zürich, Switzerland

1991

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
MASTER OF SCIENCE
May, 1992

HUMAN OPTIMAL CONTROL FOR AIRCRAFT
FLYING VARIOUS STEADY
STATE MANEUVERS

Thesis Approved:



Thesis Adviser

Dean of the Graduate College

ACKNOWLEDGMENTS

I wish to express my sincere appreciation to Dr. Peter Moretti at OSU and Dr. Torstein Fanneløp at ETHZ for making it possible for me to study at the Oklahoma State University. Many thanks go also to my thesis adviser Dr. Robert Swaim and to the members of my graduate committee Dr. Eduardo Misawa and Dr. Frank W. Chambers for the valuable help they provided.

I would also like to thank Dr. Alan J. Laub at the University of California, Santa Barbara for providing me with some efficient Fortran subroutines to solve the Riccati and the Lyapunov equation.

Ulf Nobbman provided me with the Latex Document Preparation Program on which this thesis was written. His help throughout the project and especially during the printing is fully appreciated.

I would also like to express my sincere gratitude to all my friends who helped me get through this stressful time by providing me with some distraction, so that my batteries could be recharged.

Last but not least, I extend a sincere thank you to my parents and relatives who supported my stay in the United States.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

NOMENCLATURE

A	plant matrix
a_t	tail lift curve slope
B	input matrix
b	wingspan
c	mean wing chord
E	noise matrix
g	earth acceleration of gravity
I	moment of inertia
J	performance index, Jordan block
K	solution of Riccati-equation
K_a, K_e, K_r	control system gearing constants
L	optimal feedback gain
l_0	turbulence intensity
M	Mach-number
m	aircraft mass
L, M, N	aerodynamic moments
$\Delta l, \Delta m, \Delta n$	perturbation aerodynamic moments
P, Q, R	angular velocities
p, q, r	perturbation angular velocities
Q	output weighting matrix
R	input weighting matrix
T	final time
t	time

t_0	initial time
u, v, w	perturbation velocities
u_g, v_g, w_g	gust velocities
V_H	tail volume
v_1, w_1	turbulence state variables
U, V, W	aircraft velocities
x	state vector
x_0	initial condition
y	output vector
X, Y, Z	aerodynamic forces
$\Delta x, \Delta y, \Delta z$	perturbation aerodynamic forces
W	covariance matrix of input noise
w	white noise vector

Greek:

α	angle of attack
β	sideslip angle
δ	deflection (control surface or stick)
ε	downwash angle at tail
$\eta_{u_g}, \eta_{v_g}, \eta_{w_g}$	turbulence model input
λ	eigenvalues
$\Lambda_{c/4}$	wing sweep at quarter chord line
ξ	random initial value
Σ	covariance matrix
σ_0	turbulence intensity
τ	time delay
Φ	
Φ, Θ, Ψ	Euler angle
ϕ, θ, ψ	perturbation Euler angle

Subscripts:

0	steady state
e	elevator
a	aileron
r	rudder
s	stick

Superscripts:

T	transpose matrix
$\cdot$	derivative with respect to time

CHAPTER I

INTRODUCTION

For various aircraft simulation problems, the behavior of the pilot must be simulated. For example, often one is interested in how changes in the dynamics of an airplane affect its handling qualities. There are different methods to do this. One can build a new airplane with the 'desired' properties and test fly it. It is obvious that this is not a very practical solution. A better way would be to incorporate the new dynamics in a simulator and have a test pilot assess the handling qualities. This also is a very expensive and time consuming method, which is not suitable for a quick evaluation during the design phase, for example. It would be interesting for engineers to have a computer model of the behavior of the pilot which could relate the dynamics of the aircraft to a handling quality index (Cooper-Harper scale [1]). The present work deals with such a pilot model and how a particular performance index varies as function of the maneuver flown.

Previous Work in the Field

Various pilot models have been proposed [2–4]. They can be put into two categories: the cross-over models and the optimum controller models. The latter uses the linear Gaussian optimum control theory modified to include the human's inherent limitations [4]. It is based on the assumption that a well trained pilot operates in an optimal manner based on his limited perception of the airplane state. Figure 1 shows a block diagram of such an optimal control model (OCM).

The aircraft is subjected to a random input due to the atmospheric turbulence and the pilot's control input. The pilot knows the state of the aircraft through instruments which show him various variables. But, due to instrument

reading errors (observation noise v_y) and time delays (τ) caused by the pilot's reaction time, the pilot has only a limited knowledge of the airplane state. Based on this perceived state, he decides upon a control action u_c. This control input is altered by the motor noise v_{u_c} and the neuromuscular system to give the actual aircraft control input u.

The pilot controls the aircraft so as to minimize the performance index

$$J(u) = \lim_{T\to\infty} \mathrm{E}\left\{\frac{1}{T}\int_0^T (y^{\mathrm{T}}Qy + u^{\mathrm{T}}Ru)\,dt\right\} \tag{1}$$

From the optimum control theory, the optimal feedback gain for the closed loop system can be calculated. The airplane dynamics must be linearized. Usually this is done about a horizontal steady flight condition.

Scope of this Work

The goal of this work is to find a method to calculate the value of the performance index for systems with time delays and noisy inputs and to apply it to various steady state maneuvers. Since the performance index as defined in equation (1) can be interpreted as a measurement of the difficulty of the control task, it could possibly give some indications on the handling qualities of an aircraft with the given dynamics for a particular maneuver.

CHAPTER II

PROBLEM DESCRIPTION

The pilot can be modeled as an optimal controller acting on his limited perception of the airplane state. These limitations are the observation noise due to the inacurate reading of the instruments and instrument scanning, motor noise due to the muscular control, and the pilot's reaction time delay. Figure 1 shows

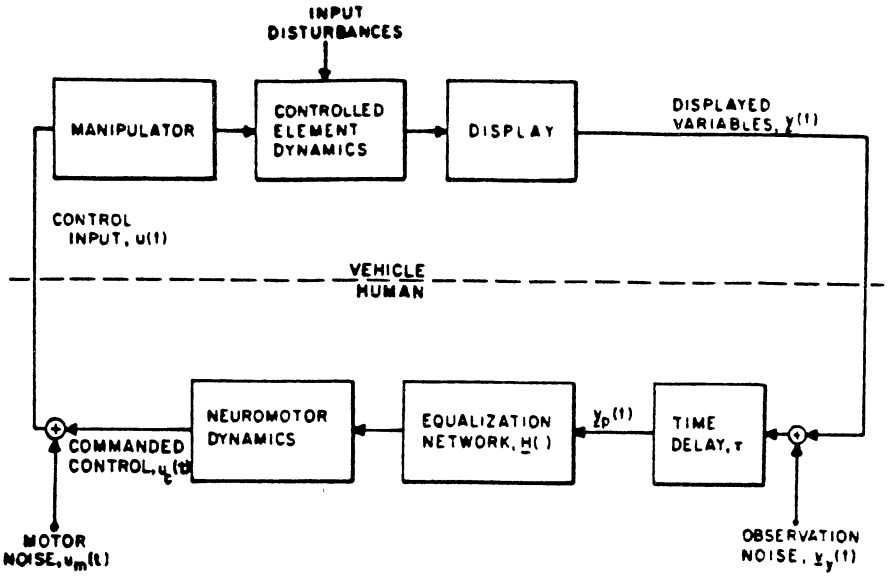


Figure 1. Optimal Pilot Model

a block diagram of the optimal control model (OCM). The observation noise and the motor noise are neglected in this work. The state equations of the linearized plant are:

$$\dot{x}(t) = Ax(t) + Bu(t) + Ew(t) \tag{2}$$

$$y(t) = Cx(t-\tau) \tag{3}$$

where x is the airplane state vector, y is the output vector, u is the control vector, w is the noise vector and τ is the time delay. The state matrix A contains the dynamics of the airplane linearized about a specified steady state maneuver. The optimal control model is based on the assumption that the pilot minimizes the value of a performance index, based on his limited knowledge of the states. The performance index is given by:

$$J(u) = \lim_{T\to\infty} \mathrm{E}\left\{\frac{1}{T}\int_0^T (x^{\mathrm{T}}Qx + u^{\mathrm{T}}Ru)\,dt\right\} \tag{4}$$

The weighting matrices R and Q must be positive definite and positive semi-definite, respectively. From the optimal control theory, we get the optimal feedback gain:

$$u_{opt} = -Lx = -R^{-1}B^{\mathrm{T}}K \tag{5}$$

with K being the solution of the algebraic Riccati equation

$$-KA - A^{\mathrm{T}}K + KBR^{-1}B^{\mathrm{T}}K - Q = 0 \tag{6}$$

From the known optimal feedback gain, the value of the performance index can be calculated. This value can then be compared for various maneuvers.

CHAPTER III

EQUATIONS OF MOTION

Aircraft Dynamics

The six rigid-body equations of motion and the three Euler angles kinematic equations are given by [5]:

$$X - mg\sin\Theta = m(\dot{U} + QW - RV) \tag{1}$$

$$Y + mg\cos\Theta\sin\Phi = m(\dot{V} + RU - PW) \tag{2}$$

$$Z + mg\cos\Theta\cos\Phi = m(\dot{W} + PV - QU) \tag{3}$$

$$L = \dot{P}I_x - QR(I_z - I_y) - (\dot{R} + PQ)I_{xz} - (\dot{Q} - PR)I_{xy} - (Q^2 - R^2)I_{yz} \tag{4}$$

$$M = \dot{Q}I_y - PR(I_x - I_z) - (\dot{P} + RQ)I_{xy} - (\dot{R} - PQ)I_{yz} - (R^2 - P^2)I_{xz} \tag{5}$$

$$N = \dot{R}I_z - PQ(I_y - I_x) - (\dot{Q} + PR)I_{yz} - (\dot{P} - QR)I_{xz} - (P^2 - Q^2)I_{xy} \tag{6}$$

$$\dot{\Phi} = P + Q\tan\Theta\sin\Phi + R\tan\Theta\cos\Phi \tag{7}$$

$$\dot{\Theta} = Q\cos\Phi - R\sin\Phi \tag{8}$$

$$\dot{\Psi} = \frac{1}{\cos\Theta}(R\cos\Phi + Q\sin\Phi) \tag{9}$$

where X,Y,Z are the aerodynamic forces and L,M,N the aerodynamic moments in the x-, y-, z-directions, respectively. U, V, W are the velocities and P, Q, R are the angular rates with respect to the body axes (see Fig. 2).

Linearizaton of the Equations

These non linear equations must be linearized so that they can be used for the pilot model. This is done by assuming that there are only small perturbations around a steady state flight condition. We define

$$\begin{array}{lll} U = U_0 + u & V = V_0 + v & W = W_0 + w \\ P = P_0 + p & Q = Q_0 + q & R = R_0 + r \\ \Theta = \Theta_0 + \theta & \Phi = \Phi_0 + \phi & \Psi = \Psi_0 + \psi \\ X = X_0 + \Delta x & Y = Y_0 + \Delta y & Z = Z_0 + \Delta z \\ L = L_0 + \Delta l & M = M_0 + \Delta m & N = N_0 + \Delta n \end{array} \tag{10}$$

where the subscript 0 denotes the steady state condition. Substituting the above terms into the equations (1-9) and neglecting the products of the perturbation terms, making small angle approximations for the sine, cosine and tangent gives the steady-state equations (11-19) and the perturbation equations (21-28).

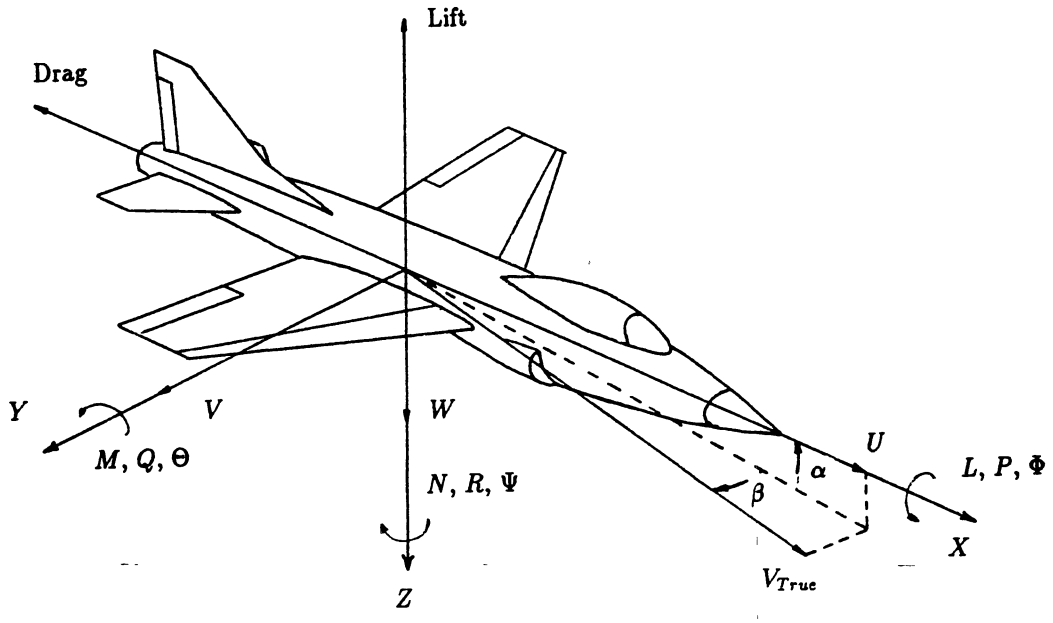


Figure 2. Airplane Coordinate System

$$m(W_0Q_0 - V_0R_0 + g\sin\Theta_0) = X_0 \tag{11}$$

$$m(U_0R_0 - W_0P_0 - g\cos\Theta_0\sin\Phi_0) = Y_0 \tag{12}$$

$$m(V_0P_0 - U_0Q_0 - g\cos\Theta_0\cos\Phi_0) = Z_0 \tag{13}$$

$$Q_0R_0(I_z - I_y) - P_0Q_0I_{xz} + P_0R_0I_{xy} + (R_0^2 - Q_0^2)I_{yz} = L_0 \tag{14}$$

$$P_0R_0(I_x - I_z) - Q_0R_0I_{xy} + P_0Q_0I_{yz} + (P_0^2 - R_0^2)I_{xz} = M_0 \tag{15}$$

$$P_0Q_0(I_y - I_x - P_0R_0I_{yz} + Q_0R_0I_{xz}(Q_0^2 - P_0^2)I_{xy}) = N_0 \tag{16}$$

$$\dot{\Phi}_0 = P_0 + (Q_0\sin\Phi_0 + R_0\cos\Phi_0)\tan\Theta_0 \tag{17}$$

$$\dot{\Theta}_0 = Q_0\cos\Phi_0 - R_0\sin\Phi_0 \tag{18}$$

$$\dot{\Psi}_0 = (Q_0\sin\Phi_0 + R_0\cos\Phi_0)\sec\Theta_0 \tag{19}$$

where X_0, Y_0, Z_0, L_0, M_0, N_0, are the steady-state aerodynamic forces and moments.

$$\Delta x = m[\dot{u} + W_0q + Q_0w - V_0r - R_0v + (g\cos\Theta_0)\theta] \tag{20}$$

$$\Delta y = m[\dot{v} + U_0r + R_0u - W_0p - P_0w + (g\cos\Theta_0\cos\Phi_0)\phi + (g\sin\Theta_0\sin\Phi 0)\theta] \tag{21}$$

$$\Delta z = m[\dot{w} + V_0p + P_0v - U_0q - Q_0u + (g\cos\Theta_0\cos\Phi_0)\phi + (g\sin\Theta_0\sin\Phi 0)\theta] \tag{22}$$

$$\Delta l = \dot{p}I_x + (Q_0r + R_0q)(I_z - I_y) - (\dot{r} + Q_0p + P_0q)I_{xz} - (\dot{q} - P_0r - R_0p)I_{xy} - 2(Q_0q - R_0r)I_{yz} \tag{23}$$

$$\Delta m = \dot{q}I_y + (P_0r + R_0p)(I_x - I_z) - (\dot{p} + Q_0r + R_0q)I_{xy} - (\dot{r} - P_0q - Q_0p)I_{yz} - 2(R_0r - P_0p)I_{xz} \tag{24}$$

$$\Delta n = \dot{r}I_z + (P_0q + Q_0p)(I_y - I_x) - (\dot{p} - Q_0r + R_0q)I_{xz} - (\dot{q} + P_0r - R_0q)I_{yz} - 2(P_0p - Q_0q)I_{xy} \tag{25}$$

$$\dot{\phi} = p + \phi(Q_0\cos\Phi_0 - R_0\sin\Phi_0)\tan\Theta_0 + q\sin\Phi_0\tan\Theta_0 + r\cos\Phi_0\tan\Theta_0 \tag{26}$$

$$\dot{\theta} = q\cos\Phi_0 - r\sin\Phi_0 - \phi(Q_0\sin\Phi_0 + R_0\cos\Phi_0) \quad (27)$$

$$\dot{\psi} = q\sin\Phi_0\sec\Theta_0 + r\cos\Phi_0\sec\Theta_0 + \phi(Q_0\cos\Phi_0 - R\sin\Phi_0)\sec\Theta_0 + \theta(Q_0\sin\Phi + R_0\cos\Phi_0)\tan\Theta_0\sec\Theta_0 \quad (28)$$

These equations are now linear with respect to the perturbation variables. The steady state maneuver can be given by specifying P_0, Q_0, R_0, U_0, V_0 and W_0.

The aircraft is assumed to be symmetrical with respect to the vertical plane. Therefore the equations can be further simplified by setting the mass moment of inertia I_{yz} and I_{xy} equal to zero. Since the angle of attack α and the sideslip angle β can be assumed to be small, they can be written as :

$$\alpha = \tfrac{w}{U_0} \text{ and } \beta = \tfrac{v}{U_0}$$

In the state variable form we have now

$$\frac{\dot{u}}{U_0} = -Q_0\alpha + R_0\beta - (\frac{g}{U_0}\cos\Theta_0)\theta + \frac{\Delta x}{mU_0} \quad (29)$$

$$\dot{\beta} = -R_0\frac{u}{U_0} + P_0\alpha - r + \frac{g}{U_0}[(\cos\Theta_0\cos\Phi_0)\phi - (\sin\Theta_0\sin\Phi_0)\theta] + \frac{\Delta y}{mU_0} \quad (30)$$

$$\dot{\alpha} = -P_0\beta - Q_0\frac{u}{U_0} - q - \frac{g}{U_0}[(\cos\Theta_0\cos\Phi_0)\phi + (\sin\Theta_0\sin\Phi_0)\theta] + \frac{\Delta z}{mU_0} \quad (31)$$

$$\dot{p} = \frac{1}{I_x}(Q_0I_{xz})p + \frac{1}{I_x}[P_0I_{xz} - R_0(I_z - I_y)]q - \frac{1}{I_x}[Q_0(I_z - I_y)]r + \frac{\Delta l}{I_x} \quad (32)$$

$$\dot{q} = -\frac{1}{I_y}(2P_0I_{xy} + R_0(I_x - I_z)]p + \frac{1}{I_y}(R_0I_{xy})q + \frac{1}{I_y}[2R_0I_{xz} - P_0(I_x - I_z)]r + \frac{\Delta n}{I_y} \quad (33)$$

$$\dot{r} = -\frac{1}{I_z}[Q_0(I_y - I_x)]p - \frac{1}{I_z}[R_0I_{xz} + P_0(I_y - I_x)]q - \frac{1}{I_z}(Q_0I_{xz} - P_0I_{yz})r + \frac{\Delta n}{I_z} \quad (34)$$

Aerodynamic Forces

The perturbation aerodynamic terms Δx, Δy, Δz, Δl, Δm and Δn can be expressed as functions of stability derivatives. The usual simplifications (neglecting stability derivatives which are small) are made. The perturbation aerodynamic forces and moments are as follows [6]:

$$\begin{aligned}
\Delta x &= \frac{1}{2}\rho U_0^2 S\left\{-[C_{D_u}+2(C_D)_0]\frac{u}{U_0}+[-C_{D_\alpha}+(C_L)_0]\alpha-C_{D_{\delta_e}}\delta_e\right. \\
&\qquad \left.+[C_{D_u}+2(C_D)_0]\frac{u_g}{U_0}-[-C_{D_\alpha}+(C_L)_0]\frac{w_g}{U_0}\right\} \qquad (35)\\
\Delta y &= \frac{1}{2}\rho U_0^2 S\left\{C_{y_\beta}\beta+C_{y_p}\left(\frac{pb}{2U_0}\right)+C_{y_r}\left(\frac{rb}{2U_0}\right)\right. \\
&\qquad \left.+C_{y_{\delta_a}}\delta_a+C_{y_{\delta_r}}\delta_r-C_{y_\beta}\frac{v_g}{U_0}\right\} \qquad (36)\\
\Delta z &= \frac{1}{2}\rho U_0^2 S\left\{-[C_{L_u}+2(C_L)_0]\frac{u}{U_0}-[C_{L_\alpha}+(C_D)_0]\alpha-C_{L_{\dot\alpha}}\left(\frac{\dot\alpha_c}{2U_0}\right)\right. \\
&\qquad -C_{L_q}\left(\frac{q_c}{2U_0}\right)-C_{L_{\delta_e}}\delta_e+[C_{L_u}+2(C_L)_0]\frac{u_g}{U_0} \\
&\qquad \left.+[C_{L_\alpha}+(C_D)_0]\frac{w_g}{U_0}\right\} \qquad (37)\\
\Delta l &= \frac{1}{2}\rho U_0^2 Sb\left\{C_{l_\beta}\beta+C_{l_p}\left(\frac{pb}{2U_0}\right)+C_{l_r}\left(\frac{rb}{2U_0}\right)\right. \\
&\qquad \left.+C_{l_{\delta_a}}\delta_a+C_{l_{\delta_r}}\delta_r-C_{l_\beta}\frac{v_g}{U_0}\right\} \qquad (38)\\
\Delta m &= \frac{1}{2}\rho U_0^2 Sbc\left\{[C_{m_u}+2(C_{m_0})_0]\frac{u}{U_0}+C_{m_\alpha}\alpha+C_{m_{\dot\alpha}}\left(\frac{\dot\alpha c}{2U_0}\right)\right. \\
&\qquad +C_{m_q}\left(\frac{qc}{2U_0}\right)+C_{m_{\delta_e}}\delta_e-[C_{m_u}+2(C_m)_0]\frac{u_g}{U_0} \\
&\qquad \left.-C_{m_\alpha}\frac{w_g}{U_0}\right\} \qquad (39)\\
\Delta n &= \frac{1}{2}\rho U_0^2 Sb\left\{C_{n_\beta}\beta+C_{n_p}\left(\frac{pb}{2U_0}\right)+C_{n_r}\left(\frac{rb}{2U_0}\right)+C_{n_{\delta_a}}\delta_a\right. \\
&\qquad \left.+C_{n_{\delta_r}}\delta_r-C_{n_\beta}\frac{v_g}{U_0}\right\} \qquad (40)
\end{aligned}$$

u_g, v_g and w_g are the components of the gust velocities acting at the center of gravity of the airplane.

Steady State Maneuver

Any steady-state maneuver is given by equations (11-19). Very often these equations can be simplified. In the following let us consider the three steady-state maneuvers:

1. unaccelerated horizontal flight
2. steady turn
3. symmetrical pullup

Unaccelerated Horizontal Flight

All the derivatives with respect to time, the angular velocities P_0, Q_0, R_0 and the velocities V_0, W_0 are zero. The equations (11-19) then reduce to

$$\begin{aligned} X_0 &= Y_0 = 0 \\ Z_0 &= -mg \\ M_0 &= N_0 = L_0 = 0 \\ P_0 &= Q_0 = R_0 = 0 \\ \Psi_0 &= \Theta_0 = \Psi_0 = 0 \end{aligned} \tag{41}$$

Steady Turn

In a steady turn, the rate of turn $\dot{\Psi}$ is constant. The other time derivatives are zero. So is the pitch angle Θ_0 and, for a coordinated turn, the side force Y_0. From equation (17), we get that P_0 must also be zero. Even for high bank angles, the rate of turn $\dot{\Psi}_0$ is small. Therefore, the products of Q_0 and R_0 can be neglected so that:

$$\begin{aligned} L_0 &\cong 0 \\ M_0 &\cong 0 \\ N_0 &\cong 0 \end{aligned} \tag{42}$$

From the side-force equation (12) we get:

$$\sin \Phi_0 = \frac{U_0 R_0}{g} \tag{43}$$

from equation (18)

$$\tan \Phi_0 = \frac{Q_0}{R_0} \tag{44}$$

and from equation (18) and (19)

$$Q_0 = \Psi_0 \sin \Phi_0 \tag{45}$$

$$R_0 = \Psi_0 \cos \Phi_0 \tag{46}$$

The load factor n is

$$n = \frac{1}{\cos \Phi_0} \tag{47}$$

By specifying any two of the variables U_0, $\dot{\Psi}$, Φ_0 or n, the steady level turn is specified.

Symmetrical Pull-up

In a steady symmetrical pull-up, P_0, R_0, Ψ_0, Φ_0 and the velocities V_0, W_0 are zero. The time derivatives are also zero except for $\dot{\Theta}_0$, which is constant. The equations for this maneuver are as follows:

$$\begin{aligned} X &= mg \sin \Theta_0 \\ Y &= 0 \\ Z &= -mg \cos \Theta_0 - U_0 Q_0 \\ L_0 &= M_0 = N_0 = 0 \\ \dot{\Theta}_0 &= \text{cst} \end{aligned} \tag{48}$$

We see that actually a symmetrical pull-up is not a true steady state maneuver since the z-force is a function of Θ_0, which changes with time. To avoid the problems due to this fact, we consider the maneuver only during a short instant of time during which Θ_0 can be assumed to remain constant.

Actuator Dynamics

The dynamics of the flight control system itself (relation between the stick deflection and the control surface deflection) can be modeled with first order differential equations:

$$\dot{\delta}_e = -\frac{1}{\tau}\delta_e + \frac{K_e}{\tau}\delta_{e_s} \tag{49}$$

$$\dot{\delta}_a = -\frac{1}{\tau}\delta_a + \frac{K_a}{\tau}\delta_{a_s} \tag{50}$$

$$\dot{\delta}_r = -\frac{1}{\tau}\delta_r + \frac{K_r}{\tau}\delta_{r_s} \tag{51}$$

where δ_e, δ_a, δ_r are the elevator, aileron and rudder deflections, respectively, and δ_{e_s}, δ_{a_s}, δ_{r_s} are the corresponding stick deflections. The actuator time constant is typically taken to be 0.05 sec., and K_e, K_a, K_r are gearing constants depending on the airplane.

CHAPTER IV

TURBULENCE MODEL

The airplane is flying in a turbulent atmosphere. Gusts give random deviations from the steady state maneuver flown, which the pilot tries to correct as good as he can.

In this work, the isotropic Dryden spectral density form is used to describe the gust components u_g, v_g, w_g. Transformation to the time domain yields five first order differential equations excited by Gaussian white noise (see [6]).

$$\dot{u}_g = -\frac{U_0}{l_0}u_g + \sigma_0\sqrt{\frac{2U_0}{\pi l_0}}\eta_{u_g} \tag{52}$$

$$\dot{v}_g = \left(\frac{U_0}{2l_0}\right)^2 v_1 + \frac{\sigma_0\sqrt{3}}{2}\sqrt{\frac{2U_0}{\pi l_0}}\eta_{v_g} \tag{53}$$

$$\dot{v}_1 = -\frac{U_0}{l_0}v_1 - v_g + \left(1 - \sigma_0\sqrt{3}\frac{l_0}{U_0}\sqrt{\frac{2U_0}{\pi l_0}}\right)\eta_{v_g} \tag{54}$$

$$\dot{w}_g = \left(\frac{U_0}{2l_0}\right)^2 w_1 + \frac{\sigma_0\sqrt{3}}{2}\sqrt{\frac{2U_0}{\pi l_0}}\eta_{w_g} \tag{55}$$

$$\dot{w}_1 = -\frac{U_0}{l_0}w_1 - w_g + \left(1 - \sigma_0\sqrt{3}\frac{l_0}{U_0}\sqrt{\frac{2U_0}{\pi l_0}}\right)\eta_{w_g} \tag{56}$$

σ_0 is the turbulence intensity and can be taken to be 6 ft/s, representing a moderate level of turbulence. A good value for the integral scale of the turbulence l_0 is 1750 ft. The input variables η_{u_g}, η_{v_g},η_{w_g} are generated by Gaussian white noise.

CHAPTER V

PLANT MATRIX

The linearized aircraft dynamics, the turbulence model and the actuator dynamics are combined together to form the augmented plant matrix A. The dynamics of the augmented system are given by the vector differential equation:

$$\dot{x}(t) = Ax(t) + Bu(t) + E\eta(t) \tag{57}$$

where the 16^{th} order state vector is defined as:

$$x(t) = \left(\frac{u}{U_0}, \beta, \alpha, p, q, r, \delta_e, \delta_a, \delta_r, \phi, \theta, u_g, v_g, v_1, w_g, w_1\right)^{\mathrm{T}} \tag{58}$$

the control vector is:

$$u(t) = (\delta_{e_s}, \delta_{a_s}, \delta_{r_s})^{\mathrm{T}} \tag{59}$$

and the noise vector

$$\eta(t) = (\eta_{u_g}, \eta_{v_g}, \eta_{w_g})^{\mathrm{T}} \tag{60}$$

The equations of motion (eq. 30-34), the turbulence equations (eq. 53-56) and the actuator dynamics (eq. 49-51) are combined to yield the linear system of differential equations in terms of the state, the input, and the noise variables. The aerodynamic forces and moments are expressed as functions of the dimensional stability derivatives as defined by Roskam [7].

$$\begin{aligned}
\frac{\dot{u}}{U_0} &= X_u\frac{u}{U_0} + R_0\beta + \left(\frac{X_\alpha}{U_0} - Q_0\right)\alpha + \left(\frac{X_{\delta_e}}{U_0}\right)\delta_e - \left(\frac{g}{U_0}\cos\Theta_0\right)\theta \\
&\quad - \left(\frac{X_u}{U_0}\right)u_g - \left(\frac{X_\alpha}{U_0^2}\right)w_g
\end{aligned} \tag{61}$$

$$\begin{aligned}
\dot{\beta} &= (-R_0)\frac{u}{U_0} + \left(\frac{Y_\beta}{U_0}\right)\beta + (P_0)\alpha + \left(\frac{Y_p}{U_0}\right)p + \left(\frac{Y_r}{U_0} - 1\right)r \\
&\quad + \left(\frac{Y_{\delta_a}}{U_0}\right)\delta_a + \left(\frac{Y_{\delta_r}}{U_0}\right)\delta_r + \left(\frac{g}{U_0}\cos\Theta_0\cos\Phi_0\right)\phi \\
&\quad - \left(\frac{g}{U_0}\sin\Theta_0\sin\Phi_0\right)\theta - \left(\frac{Y_\beta}{U_0^2}\right)v_g
\end{aligned} \tag{62}$$

$$\dot{\alpha} = (Z_u + Q_0)\frac{u}{U_0} - P_0\beta + \left(\frac{Z_\alpha}{U_0}\right)\alpha + q + \left(\frac{Z_{\delta_e}}{U_0}\right)\delta_e - \left(\frac{g}{U_0}\cos\Theta_0 \sin\Phi_0\right)\phi - \left(\frac{g}{U_0}\sin\Theta_0\cos\Phi_0\right)\theta - \left(\frac{Z_u}{U_0}\right)u_g - \left(\frac{Z_\alpha}{U_0^2}\right)w_g \quad (63)$$

$$\dot{p} = (L_\beta)\beta + \left[L_p + \frac{1}{I_x}Q_0 I_{xz}\right]p + \frac{1}{I_x}\left[P_0 I_{xz} - R_0(I_z - I_y)\right]q + \left[L_r - \frac{1}{I_x}Q_0(I_z - I_y)\right]r + L_{\delta_a}\delta_a + L_{\delta_r}\delta_r - \left(\frac{L_\beta}{U_0}\right)v_g \quad (64)$$

$$\dot{q} = \left[U_0 M_0 + M_{\dot{\alpha}}(Q_0 + Z_0)\right]\frac{u}{U_0} - (M_{\dot{\alpha}}P_0)\beta + \left[M_\alpha + \frac{M_{\dot{\alpha}}Z_\alpha}{U_0}\right]\alpha - \frac{1}{I_y}\left[2P_0 I_{xz} + R_0(I_x - I_z)\right]p + \left[M_{\dot{\alpha}} + M_q\right]q + \frac{1}{I_y}\left[2R_0 I_{xz} - P_0(I_x - I_z)\right]r \left[M_{\delta_e} + \frac{M_{\dot{\alpha}}Z_{\delta_e}}{U_0}\right]\delta_e - \left[\frac{M_{\dot{\alpha}}g}{U_0}\cos\Theta_0\cos\Phi_0\right]\phi - \left[\frac{M_{\dot{\alpha}}g}{U_0}\sin\Theta_0\cos\Phi_0\right]\theta - \left[M_u + \frac{M_{\dot{\alpha}}Z_u}{U_0}\right]u_g - \frac{1}{U_0}\left[M_{\dot{\alpha}}Z_\alpha + M_\alpha\right]w_g \quad (65)$$

$$\dot{r} = (N_\beta)\beta + \left[N_p - \frac{1}{I_z}Q_0(I_y - I_x)\right]p - \frac{1}{I_z}\left[R_0 I_{xz} + P_0(I_y - I_x)\right]q + \left[N_r - \frac{1}{I_z}Q_0 I_{xz}\right]r + N_{\delta_a}\delta_a + N_{\delta_r}\delta_r - \left(\frac{N_\beta}{U_0}\right)v_g \quad (66)$$

$$\dot{\delta}_e = -\frac{1}{\tau}\delta_e + \frac{K_e}{\tau}\delta_{e_s} \quad (67)$$

$$\dot{\delta}_a = -\frac{1}{\tau}\delta_a + \frac{K_a}{\tau}\delta_{a_s} \quad (68)$$

$$\dot{\delta}_r = -\frac{1}{\tau}\delta_r + \frac{K_r}{\tau}\delta_{r_s} \quad (69)$$

$$\dot{\phi} = p + \phi(Q_0\cos\Phi_0 - R_0\sin\Phi_0)\tan\Theta_0 + q\sin\Phi_0\tan\Theta_0 + r\cos\Phi_0\tan\Theta_0 \quad (70)$$

$$\dot{\theta} = q\cos\Phi_0 - r\sin\Phi_0 - \phi(Q_0\sin\Phi_0 + R_0\cos\Phi_0) \quad (71)$$

$$\dot{u}_g = -\frac{U_0}{l_0} u_g + \sigma_0 \sqrt{\frac{2U_0}{\pi l_0}} \eta_{u_g} \tag{73}$$

$$\dot{v}_g = \left(\frac{U_0}{2l_0}\right)^2 v_1 + \frac{\sigma_0\sqrt{3}}{2} \sqrt{\frac{2U_0}{\pi l_0}} \eta_{v_g} \tag{74}$$

$$\dot{v}_1 = -\frac{U_0}{l_0} v_1 - v_g + \left(1 - \sigma_0\sqrt{3}\frac{l_0}{U_0}\sqrt{\frac{2U_0}{\pi l_0}}\right) \eta_{v_g} \tag{75}$$

$$\dot{w}_g = \left(\frac{U_0}{2l_0}\right)^2 w_1 + \frac{\sigma_0\sqrt{3}}{2} \sqrt{\frac{2U_0}{\pi l_0}} \eta_{w_g} \tag{76}$$

$$\dot{w}_1 = -\frac{U_0}{l_0} w_1 - w_g + \left(1 - \sigma_0\sqrt{3}\frac{l_0}{U_0}\sqrt{\frac{2U_0}{\pi l_0}}\right) \eta_{w_g} \tag{77}$$

Aerodynamic-, mass- and inertia-data are taken for a fighter type aircraft (Fig. 3). The stability derivatives have been determined in flight tests and are available in tabulated form (see Appendix A).

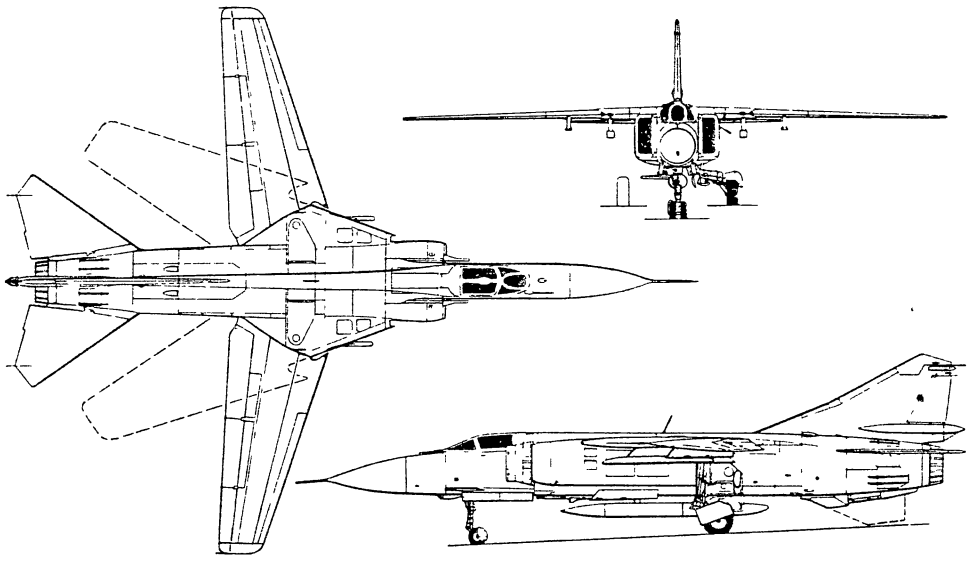


Figure 3. Geometry of the simulated aircraft

CHAPTER VI

STABILITY DERIVATIVES

The dimensional stability derivatives used in equations (62-67) are:

$X_u = -\frac{qS}{mU_0}(C_{D_u} + 2(C_D)_0)$	$[s^{-1}]$	$Y_r = \frac{qSb}{2mU_0}C_{y_r}$	$[\frac{ft}{s\,rad}]$
$X_\alpha = \frac{qS}{mU_0}(-C_{D_\alpha} + 2(C_L)_0)$	$[\frac{ft}{s^2 rad}]$	$Z_u = -\frac{qS}{mU_0}(C_{L_u} + 2(C_L)_0)$	$[s^{-1}]$
$X_{\delta_e} = (C_{D_u} + 2(C_D)_0)$	$[s^{-1}]$	$Z_\alpha = -\frac{qS}{m}(C_{L_u} + 2(C_L)_0)$	$[\frac{ft}{s^2 rad}]$
$Y_\beta = \frac{qS}{m}C_{y_\beta}$	$[\frac{ft}{s^2 rad}]$	$Z_{\delta_e} = -\frac{qS}{m}C_{L_{\delta_e}}$	$[s^{-2}]$
$Y_p = -\frac{qSb}{2mU_0}C_{y_p}$	$[\frac{ft}{srad}]$		

$L_\beta = \frac{qsb}{I_{xx}}C_{l_\beta}$	$[s^{-2}]$	$M_q = \frac{qsbc}{2I_{yy}U_0}C_{m_q}$	$[s^{-1}]$
$L_p = \frac{qsb^2}{2I_{xx}U_0}C_{l_p}$	$[s^{-1}]$	$M_{\delta_e} = \frac{qsb}{I_{yy}}C_{m_{\delta_e}}$	$[s^{-2}]$
$L_r = \frac{qsb^2}{2I_{xx}U_0}C_{l_r}$	$[s^{-1}]$	$N_\beta = \frac{qsb}{I_{xx}}C_{n_\beta}$	$[s^{-2}]$
$L_{\delta_a} = \frac{qsb}{I_{xx}}C_{l_{\delta_a}}$	$[s^{-2}]$	$N_p = \frac{qsb^2}{2I_{zz}U_0}C_{n_p}$	$[s^{-1}]$
$L_{\delta_r} = \frac{qsb}{I_{xx}}C_{l_{\delta_r}}$	$[s^{-2}]$	$N_r = \frac{qsb^2}{2I_{xx}U_0}C_{n_r}$	$[s^{-1}]$
$M_u = \frac{qsb}{I_{yy}U_0}(C_{m_u} + 2(C_m)_0)$	$[\frac{rad}{fts}]$	$N_{\delta_a} = \frac{qsb}{I_{zz}}C_{n_{\delta_a}}$	$[s^{-2}]$
$M_\alpha = \frac{qsb}{I_{yy}}C_{m_\alpha}$	$[s^{-2}]$	$N_{\delta r} = \frac{qsb}{I_{zz}}C_{n_{\delta_r}}$	$[s^{-2}]$
$M_{\dot\alpha} = \frac{qsbc}{2I_{yy}U_0}C_{m_\alpha}$	$[s^{-1}]$		

From the specified steady state maneuver, the lift coefficient $(C_L)_0$, the drag coefficient $(C_D)_0$, and the longitudinal moment coefficient $(C_m)_0$ are calculated. By interpolation in the aerodynamic data table, the non dimensional coefficients C_{L_α}, $C_{L_{\delta_e}}$, C_{y_β}, $C_{y_{\delta_r}}$, C_{l_β}, C_{l_r}, C_{l_p}, $C_{l_{\delta_r}}$, $C_{l_{\delta_a}}$, C_{m_α}, $C_{m_{\delta_e}}$, C_{m_q}, C_{n_β}, C_{n_r}, C_{n_p}, $C_{n_{\delta_r}}$, $C_{n_{\delta_a}}$ are determined as a function of the Mach number and the maneuver. $C_{D_{\delta_e}}$, C_{y_p}, C_{y_r} and $C_{y_{\delta_a}}$ can be neglected. All the other stability derivatives must be calculated or estimated from the available data.

In the data table, the drag coefficient is given as a function of the Mach number. The coefficient C_{D_M} can therefore be determined. The speed damping derivative C_{D_u} is then equal to:

$$C_{D_u} = \frac{\partial C_D}{\partial M}\frac{\partial M}{\partial (u/U_0)} = M\frac{\partial C_D}{\partial M} \tag{77}$$

The derivative C_{D_α} can be expressed as a function of $C_{D_{C_L^2}}$:

$$C_{D_{C_L^2}} = \frac{\partial C_D}{\partial C_L^2}\frac{\partial C_L^2}{\partial \alpha} = \frac{\partial C_D}{\partial C_L^2}2C_L\frac{\partial C_L}{\partial \alpha} = 2(C_L)_0 C_{L_\alpha}\frac{\partial C_D}{\partial C_L^2} \tag{78}$$

The lift derivative C_{L_u} can be written as

$$C_{L_u} = -M\frac{\partial C_{L_\alpha}}{\partial M} \tag{79}$$

For Mach numbers below 1, C_{L_M} can be estimated using the Prandtl-Glauert rule which gives the lift coefficient as a function of Mach number for two-dimensional flows:

$$C_L = \frac{C_{L_\alpha}\alpha}{\sqrt{1-M^2}} \tag{80}$$

Upon differentiation with respect to M, we get:

$$\frac{\partial C_L}{\partial M} = \frac{M}{1-M^2}(C_L)_0$$

and therefore

$$C_{L_u} = \frac{M^2}{1-M^2}(C_L)_0 \tag{81}$$

The derivative C_{m_u} can be expressed as a function of C_{m_M}

$$C_{m_u} = \frac{\partial C_m}{\partial M}\frac{\partial M}{\partial (u/U_0)} = M\frac{\partial C_m}{\partial M} \tag{82}$$

The C_{m_α} derivative is not given and can not be neglected. It is therefore necessary to estimate its value. Roskam [8] suggests the following formula:

$$C_{m_\alpha} = -2a_t V_H \frac{l_t}{c}\frac{\partial \varepsilon}{\partial \alpha} \tag{83}$$

where a_t is the tail lift curve slope, V_H is the tail volume coefficient, L_t is the tail moment arm, c is the wing mean chord and ε is the downwash angle at the tail.

This is only the contribution of the tail. The wing and the fuselage effects are neglected. Roskam gives also an approximation formula for the derivative C_{m_q}:

$$C_{m_q} = -2a_t V_H \frac{l_t}{c} \tag{84}$$

Thus, substituting equation (84) into equation (83), we get:

$$C_{m_\alpha} = C_{m_q} \frac{\partial \varepsilon}{\partial \alpha} \tag{85}$$

The derivative C_{m_q} is known. The problem is now reduced to the calculation of the change in downwash with the angle of attack. This value can be estimated using the following formula (see Roskam [8]):

$$\left.\frac{\partial \varepsilon}{\partial \alpha}\right|_M = \left.\frac{\partial \varepsilon}{\partial \alpha}\right|_0 \frac{C_{L_\alpha}(M)}{C_{L_\alpha}(M=0)} \tag{86}$$

with

$$\left.\frac{\partial \varepsilon}{\partial \alpha}\right|_{M=0} = 4.44 \left(K_A K_\lambda K_H \sqrt{\cos \Lambda_{c/4}}\right) \tag{87}$$

where $\Lambda_{c/4}$ is the sweep angle of the wing quarter chord line ($\Lambda_{c/4} \cong 66\,\text{deg}$). The constants K are only function of the geometry of the aircraft (see fig. 3) and are estimated to be

$$\begin{aligned} K_A &= \tfrac{1}{A} - \tfrac{1}{1+A^{1.7}} && \cong 0.2902 \\ K_\lambda &= \tfrac{10-3\lambda}{7} && \cong 0.5714 \\ K_H &= \frac{1-\frac{h_H}{b}}{\sqrt[3]{\frac{2l_h}{b}}} && \cong 0.989 \end{aligned}$$

CHAPTER VII

CALCULATION OF THE PERFORMANCE INDEX

As mentioned before, we are interested in calculating the performance index (Eq. 92) for a variety of steady state maneuvers. One of the difficulties is that the pilot model includes time delays. Let us first calculate the performance index for a system without time delays.

Linear System without Time Delay

Let us consider the linear system given by

$$\dot{x}(t) = Ax(t) + Bu(t) + Ew(t) \tag{88}$$

$$y(t) = Cx(t) \tag{89}$$

where x is the state vector, u is the control vector and w is the noise vector. The initial condition ξ is non-deterministic but can be described by its expected value and its covariance.

$$\begin{aligned} \mathrm{E}\{\xi\} &= x_0 \\ \mathrm{E}\{(\xi - x_0)(\xi - x_0)^{\mathrm{T}}\} &= \Sigma_0 \end{aligned} \tag{90}$$

The input noise has the following expected values:

$$\begin{aligned} \mathrm{E}\{w(t)\} &= 0 \\ \mathrm{E}\{w(t)w(\tau)^{\mathrm{T}}\} &= W\delta(t-\tau) \\ \mathrm{E}\{w(t)(\xi - x_0)^{\mathrm{T}}\} &= 0 \end{aligned} \tag{91}$$

The control input to minimize the performance index given by equation

$$J(u_{opt}) = \lim_{T\to\infty} \mathrm{E}\left\{\frac{1}{T}\int_0^T [(x-\bar{x})^{\mathrm{T}}Q(x-\bar{x}) + (u-\bar{u})^{\mathrm{T}}R(u-\bar{u})]\,dt\right\} \tag{92}$$

is

$$u_{opt} = -R^{-1}B^{\mathrm{T}}Kx = -Lx \tag{93}$$

where K is the solution of the Riccati-equation:

$$\dot{K} = -A^{\mathrm{T}}K - KA + KBR^{-1}B^{\mathrm{T}}K - Q \tag{94}$$

The closed loop state equation is then:

$$\dot{x} = (A - BL)x + Ew \tag{95}$$

averaging the above equation over time yields

$$\dot{\bar{x}} = (A - BL)\bar{x} \tag{96}$$

and substracting equation (96) from (95)

$$\begin{aligned} \dot{x} - \dot{\bar{x}} &= (A - BL)(x - \bar{x}) + Ew \\ x(t_0) - \bar{x}(t_0) &= \xi - x_0 \end{aligned} \tag{97}$$

The solution of equation (97) is

$$x - \bar{x} = \Phi(t, t_0)[\xi - x_0] + \int_{t_0}^{t} \Phi(t, \sigma)Ew(\sigma)\, d\sigma \tag{98}$$

where $\Phi(t, t_0)$ is the transiton function. The variance Σ_0 can now be calculated

$$\begin{aligned} \Sigma(t) &= \mathrm{E}\{[x(t) - \bar{x}(t)][x(t) - \bar{x}(t)]^{\mathrm{T}}\} \\ &= \mathrm{E}\{\Phi(t, t_0)[\xi - x_0](\phi(t, t_0)[\xi - x_0])^{\mathrm{T}}\} \\ &\quad + \mathrm{E}\{\Phi(t, t_0)[\xi - x_0](\int_{t_0}^{t} \Phi(t, \rho)Ew(\rho)\, d\rho)^{\mathrm{T}}\} \\ &\quad + \mathrm{E}\{(\int_{t_0}^{t} \Phi(t, \sigma)Ew(\sigma)\, d\sigma)(\Phi(t, t_0)[\xi - x_0])^{\mathrm{T}}\} \\ &\quad + \mathrm{E}\{(\int_{t_0}^{t} \Phi(t, \sigma)Ew(\sigma)\, d\sigma)(\int_{t_0}^{t} \Phi(t, \rho)Ew(\rho)\, d\rho)^{\mathrm{T}}\} \\ &= \Phi(t, t_0)\, E(\xi - x_0)(\xi - x_0)^{T}\, \Phi^{\mathrm{T}}(t, t_0) \\ &\quad + \int_{t_0}^{t}\int_{t_0}^{t} \Phi(t, \sigma)E\, \mathrm{E}\{w(\sigma)w^{\mathrm{T}}(\rho)\}\, E^{\mathrm{T}}\Phi^{\mathrm{T}}(t, \rho)\, d\sigma d\rho \\ &= \Phi(t, t_0)\Sigma_0\Phi^{\mathrm{T}}(t, t_0) + \int_{t_0}^{t}\int_{t_0}^{t} \Phi(t, \sigma)EW\delta(\sigma - \rho)E^{\mathrm{T}}\Phi^{\mathrm{T}}(t, \rho)\, d\sigma d\rho \end{aligned}$$

Finally, we get

$$\Sigma(t) = \Phi(t,t_0)\Sigma_0\Phi^{\mathrm{T}}(t,t_0) + \int_{t_0}^{t} \Phi(t,\sigma)EWE^{\mathrm{T}}\Phi^{\mathrm{T}}(t,\sigma)\,d\sigma \tag{99}$$

It can be shown that equation (99) is a solution of the differential equation

$$\begin{aligned} \dot{\Sigma}(t) &= (A-BL)\Sigma + \Sigma(A-BL)^{\mathrm{T}} + EWE^{\mathrm{T}} \\ \Sigma(t_0) &= \Sigma_0 \end{aligned} \tag{100}$$

The expected value of the control input is (using equation 93)

$$\begin{aligned} \mathrm{E}\{[u(t)-\bar{u}(t)][u(t)-\bar{u}(t)]^{\mathrm{T}}\} &= \mathrm{E}\{L[x(t)-\bar{x}(t)][x(t)-\bar{x}(t)]^{\mathrm{T}}L^{\mathrm{T}}\} \\ &= L\,\mathrm{E}\{[x(t)-\bar{x}(t)][x(t)-\bar{x}(t)]^{\mathrm{T}}\}\,L^{\mathrm{T}} \\ &= L\Sigma\mathrm{L}^{\mathrm{T}} \end{aligned} \tag{101}$$

The minimum of the performance index can now be calculated

$$\begin{aligned} J(u_{opt}) &= \lim_{T\to\infty} \mathrm{E}\left\{\frac{1}{T}\int_0^T [(x-\bar{x})^{\mathrm{T}}Q(x-\bar{x}) + (u-\bar{u})^{\mathrm{T}}R(u-\bar{u})]\,dt\right\} \\ &= \mathrm{E}\{(x-\bar{x})^{\mathrm{T}}Q(x-\bar{x}) + (u-\bar{u})^{\mathrm{T}}R(u-\bar{u})\} \\ &= \mathrm{E}\{\mathrm{tr}(Q(x-\bar{x})(x-\bar{x})^{\mathrm{T}}) + \mathrm{tr}(R(u-\bar{u})(u-\bar{u})^{\mathrm{T}})\} \\ &= \mathrm{E}\{\mathrm{tr}(Q\Sigma) + \mathrm{tr}(RL\Sigma L^{\mathrm{T}})\} \\ &= \mathrm{tr}\left(\lim_{T\to\infty} \mathrm{E}\left\{\frac{1}{T}\int_0^T (Q\Sigma + RL\Sigma L^{\mathrm{T}})\,dt\right\}\right) \end{aligned}$$

The equation is augmented by

$$\int_0^T (\dot{K}\Sigma + \dot{\Sigma}K)\,dt + \Sigma(T)K - \Sigma_0 K = 0$$

and using equation (93), (94) and (100) we get

$$J(u_{opt}) = \mathrm{tr}\left(\lim_{T\to\infty} \mathrm{E}\left\{\frac{1}{T}\int_0^T KEWE^{\mathrm{T}}\,dt\right\} + \Sigma_0 K\right)$$

If we assume that $\Sigma_0 = 0$, we get

$$J(u_{opt}) = \mathrm{tr}\left(\lim_{T\to\infty} \mathrm{E}\left\{\frac{1}{T}\int_0^T KEWE^{\mathrm{T}}\,dt\right\}\right) = \mathrm{tr}(KEWE^{\mathrm{T}}) \tag{102}$$

Linear System with Time Delay

In the optimal pilot model the output y is delayed by an amount τ. The state space model is now as follows:

$$\dot{x}(t) = Ax(t) + Bu(t) + Ee(t) \tag{103}$$

$$y(t) = Cx(t - \tau) \tag{104}$$

Let us calculate the value of the performance index

$$J(u) = \lim_{T \to \infty} \mathrm{E}\left\{ \frac{1}{T} \int_0^T (y^{\mathrm{T}} Q y + u^{\mathrm{T}} R u)\, dt \right\} \tag{105}$$

where, as usual, R and Q are the positive definite and positive semi-definite weighting matrices. Kleinman [9] has derived a solution to this problem if the control matrix C and the noise matrix E are unit matrices ($C = E = I$). The derivation is repeated here for our more general case. To be able to calculate an inverse system, the output vector is expanded. We define:

$$\tilde{y} = \begin{bmatrix} y \\ \bar{y} \end{bmatrix} \tag{106}$$

so that

$$\tilde{y}(t) = \tilde{C} x(t - \tau) \tag{107}$$

where

$$\tilde{C} = \begin{bmatrix} C \\ \bar{C} \end{bmatrix} \tag{108}$$

is a square n x n matrix. The matrix $\bar{C}$ is chosen so that $\tilde{C}$ can be inverted. We left multiply equation (103) by the matrix $\tilde{C}$ and substitute equation (107). We get:

$$\dot{\tilde{y}} = Zy(t) + \tilde{C}Bu(t - \tau) + \tilde{C}Ew(t - \tau) \tag{109}$$

where $Z = \tilde{C}A\tilde{C}^{-1}$ We define the deterministic contribution to the output of the input u:

$$\dot{\tilde{y}}_u = Zy(t) + \tilde{C}Bu(t - \tau) \tag{110}$$

Let r be the stochastic process defined by

$$r = \tilde{y} - \tilde{y}_u \tag{111}$$

r is then solution of the differential equation

$$\dot{r} = Zr(t) + \tilde{C}Ew(t-\tau) \tag{112}$$

The expected value $\hat{x}$ of x is

$$\tilde{C}\hat{x}(t) = \tilde{y}_u(t+\tau) + \mathrm{E}r(t+\tau) \tag{113}$$

And, since r is the output of a linear system with the Gaussian noise $w(t-\tau)$ as input, the equation (113) becomes:

$$\tilde{C}\hat{x}(t) = \tilde{y}_u(t+\tau) + \mathrm{e}^{Z\tau}r(t) \tag{114}$$

From equations (110), (112) and (114) we get:

$$\dot{\hat{x}}(t) = A\hat{x} + Bu(t) + \tilde{C}^{-1}\mathrm{e}^{Z\tau}\tilde{C}Ew(t-\tau) \tag{115}$$

The prediction error is:

$$\tilde{C}e(t) = \tilde{C}x(t) - \tilde{C}\hat{x} = \tilde{y}(t+\tau) - \tilde{y}_u(t+\tau) - \mathrm{e}^{Z\tau} = r(t+\tau) - \mathrm{e}^{Z\tau}r(t) \tag{116}$$

The above equation can be written as an integral:

$$\begin{aligned}
\tilde{C}e(t) &= \mathrm{e}^{Z(t+\tau-\sigma)}r(\sigma)\Big|_t^{t+\tau} \\
&= \int_t^{t+\tau} \frac{\partial}{\partial\sigma}\tilde{C}e(t)\,d\sigma \\
&= \int_t^{t+\tau} \left[-Z\mathrm{e}^{(t+\tau-\sigma)}r(\sigma) + \mathrm{e}^{Z(t+\tau-\sigma)}\dot{r}(\sigma)\right]\,d\sigma \\
&= \int_t^{t+tau} \left[-Z\mathrm{e}^{(t+\tau-\sigma)}r(\sigma) + \mathrm{e}^{Z(t+\tau-\sigma)}(Zr(\sigma) + \tilde{C}Ew(t-\tau))\right]\,d\sigma \\
&= \int_t^{t+\tau} \mathrm{e}^{Z(t+\tau-\sigma)}\tilde{C}Ew(t-\tau)\,d\sigma \\
\tilde{C}e(t) &= \int_0^{\tau} \mathrm{e}^{Z\zeta}\tilde{C}Ew(t-\zeta)\,d\zeta
\end{aligned}$$

The covariance of the error is then:

$$\mathrm{E}\{\tilde{C}e(t)(\tilde{C}e(t))^{\mathrm{T}}\} = \int_0^{\tau} \mathrm{e}^{Z\zeta}\tilde{C}EWE^{\mathrm{T}}\tilde{C}^{\mathrm{T}}\mathrm{e}^{Z^{\mathrm{T}}\zeta}\,d\zeta \tag{117}$$

With $\tilde{y}(t) = \tilde{C}\hat{x}(t) + \tilde{C}e(t)$, the expected value of $y^{\mathrm{T}}Qy$ becomes:

$$\mathrm{E}\{y^{\mathrm{T}}Qy\} = \mathrm{E}\{\tilde{y}^{\mathrm{T}}\tilde{Q}\tilde{y}\} = \mathrm{E}\{\hat{x}^{\mathrm{T}}\tilde{C}^{\mathrm{T}}\tilde{Q}\tilde{C}\hat{x} + 2\hat{x}^{\mathrm{T}}\tilde{C}^{\mathrm{T}}\tilde{Q}\tilde{C}e + \mathrm{e}^{\mathrm{T}}\tilde{C}^{\mathrm{T}}\tilde{Q}\tilde{C}e\} \tag{118}$$

where

$$\tilde{Q} = \begin{bmatrix} Q & 0 \\ 0 & 0 \end{bmatrix} \tag{119}$$

The second term is zero since e and $\hat{x}$ are not correlated. The performance index (eq. 105) is then

$$J(u) = \lim_{T\to\infty} \mathrm{E}\left\{\frac{1}{T}\int_0^T (\hat{x}^{\mathrm{T}}Q^*\hat{x} + \mathrm{e}^{\mathrm{T}}Q^*e + u^{\mathrm{T}}Ru)\,dt\right\} \tag{120}$$

with $Q^* = \tilde{C}^{\mathrm{T}}\tilde{Q}\tilde{C}$ Since the error e is independent of the control vector u, it can be taken out of the integral:

$$J(u) = \mathrm{E}\{\mathrm{e}^{\mathrm{T}}Q^*e\} + \lim_{T\to\infty} \mathrm{E}\left\{\frac{1}{T}\int_0^T (\hat{x}^{\mathrm{T}}Q^*\hat{x} + u^{\mathrm{T}}Ru)\,dt\right\} \tag{121}$$

The second part of the performance index (eq. 121) is similar to the problem without time delay (see page 20). The optimum value of the performance index is then

$$J(u_{opt}) = \mathrm{tr}[\tilde{C}^{\mathrm{T}}\tilde{Q}(\int_0^{\tau} \mathrm{e}^{Z\zeta}\tilde{C}EWE^{\mathrm{T}}\tilde{C}^{\mathrm{T}}\mathrm{e}^{Z^{\mathrm{T}}\zeta}\,d\zeta)\tilde{C}^{-T}] + \mathrm{tr}(K\tilde{E}W\tilde{E}^{\mathrm{T}}) \tag{122}$$

where $\tilde{C}\tilde{E} = \mathrm{e}^{Z\tau}\tilde{C}E$.

Solving the Integral

In the calculation of the performance index of systems with time delays (eq. 122), an integral of the form

$$\mathbf{I} = \int_0^T \mathrm{e}^{A\sigma} W \mathrm{e}^{A^T\sigma}\, d\sigma \tag{123}$$

must be solved. Performing the integration numerically over time is not an efficient way to solve the problem. A better method is presented here.

The n x n A-matrix is decomposed into its Jordan form:

$$A = QJQ^{-1} \tag{124}$$

$$Q = [v_1, v_2, \cdots, v_n] \tag{125}$$

$$J = \begin{bmatrix} J_1 & 0 & \cdots & 0 \\ 0 & J_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & J_M \end{bmatrix} \tag{126}$$

where v_I are the generalized eigenvectors of the matrix A and J_I are Jordan blocks

$$J_I = \begin{bmatrix} \lambda_I & 1 & 0 & \cdots & 0 \\ 0 & \lambda_I & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & & \ddots & \ddots & 1 \\ 0 & \cdots & \cdots & 0 & \lambda_I \end{bmatrix} \tag{127}$$

From linear algebra we have

$$\mathrm{e}^{At} = Q\mathrm{e}^{Jt}Q^{-1} \tag{128}$$

We define the elements of e^{Jt} so that the submatrices K_I have same dimensions as the blocks J_I.

$$\mathrm{e}^{Jt} = \begin{bmatrix} K_1 & 0 & \cdots & 0 \\ 0 & K_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & K_M \end{bmatrix} \tag{129}$$

From the properties of the Jordan decomposition, the elements of the n x n submatrix K_I are given by

$$K_I = \begin{bmatrix} 1 & t & \frac{1}{2}t^2 & \cdots & \frac{t^{n-1}}{(n-1)!} \\ 0 & 1 & t & \cdots & \frac{t^{n-2}}{(n-2)!} \\ \vdots & 0 & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & t \\ 0 & \cdots & \cdots & 0 & 1 \end{bmatrix} \tag{130}$$

Substituting equation (128) into equation (123), we get for the integrand

$$e^{At}We^{A^Tt} = Qe^{Jt}\tilde{W}e^{J^Tt}Q^T \tag{131}$$

where $W = Q\tilde{W}Q^T$

The matrix Q is not a function of time and can be taken out of the integral. We now have to solve

$$\mathbf{I_D} = \int_0^T e^{J\sigma}\tilde{W}e^{J^T\sigma}\, d\sigma \tag{132}$$

The elements of the integrand in equation (132) are

$$e^{Jt}\tilde{W}e^{J^Tt} = \begin{bmatrix} K_1\tilde{W}_{11}K_1^T & K_1\tilde{W}_{12}K_2^T & \cdots & K_1\tilde{W}_{1M}K_M^T \\ K_2\tilde{W}_{21}K_1^T & K_2\tilde{W}_{22}K_2^T & \cdots & K_2\tilde{W}_{2M}K_M^T \\ \vdots & \vdots & \ddots & \vdots \\ K_M\tilde{W}_{M0}K_1^T & K_M\tilde{W}_{M2}K_2^T & \cdots & K_M\tilde{W}_{MM}K_M^T \end{bmatrix} \tag{133}$$

The elements of the submatrices $K_I\tilde{W}_{IJ}K_J^T$ are

$$K_I\tilde{W}_{IJ}K_J^T = e^{(\lambda_I+\lambda_J)t}\sum_{l=j}^{m}\sum_{k=i}^{n}(\tilde{W}_{IJ})_{k,l}\frac{t^{k-i}}{(k-i)!}\frac{t^{l-j}}{(l-j)!} \tag{134}$$

with $K_{I_{n\,x\,n}}$, $K_{J_{m\,x\,m}}$, $i \leq n$ and $j \geq m$.

The integration over time can now be performed analytically

$$\int_0^T t^q e^{\alpha t}\, dt = e^{\alpha T}\sum_{i=0}^{q}(-1)^i\frac{T^{q-i}}{\alpha^{i+1}}\frac{T^{q-p}}{(q-i)!} - \frac{(-1)^q q!}{\alpha^{q+1}} \tag{135}$$

Now, combining equations (134) and (135) we get

$$\begin{aligned}(\mathbf{I_{D_{I,J}}})_{i,j} &= \int_0^T (K_I \tilde{W}_{IJ} K_J^T)_{i,j}\, dt \\ &= \sum_{l=j}^{m}\sum_{k=i}^{n} \frac{(W_{IJ})_{k,l}}{(k-i)!(l-j)!}\left(\sum_{p=0}^{k+l-i-j=q} \frac{(-1)^p q!}{\alpha^{p+1}} \frac{T^{q-p}}{(q-p)!} e^{\alpha T} - \frac{(-1)^q q!}{\alpha^{q+1}}\right)\end{aligned} \tag{136}$$

with $\alpha = \lambda_I + \lambda_J \neq 0$.

If all Jordan blocks have dimension 1, i.e. M=N, equation (136) reduces to

$$(\mathbf{I_{D_{I,J}}})_{i,j} = \frac{\tilde{W}_{IJ}}{\lambda_I + \lambda_J}(e^{(\lambda_I+\lambda_J)T} - 1) \tag{137}$$

The matrix I_D calculated using either equation (136) or (137) must now be left-multiplied by Q and right-multiplied by Q^T to get the result of the integral of equation (123).

$$\mathbf{I} = Q\mathbf{I_D}Q^T \tag{138}$$

Jordan Decomposition of the Plant Matrix

The plant matrix in our problem can be written as

$$A = \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix} \tag{139}$$

A_{11} contains the dynamics of the aircraft, A_{22} contains the dynamics of the first order Dryden turbulence model and A_{12} is the coupling matrix between the airplane dynamics and the turbulence. A_{22} is not dependent on the maneuver flown. The jordan decomposition of A_{22} can be found algebraically.

$$A_{22} = QJQ^{-1} \tag{140}$$

On the matrix A_{11}, an eigendecomposition can be performed so that

$$A_{11} = P\Lambda P^{-1} \tag{141}$$

where Λ is a diagonal matrix containing the eigenvalues of A_{11} and P contains the corresponding eigenvectors.

The plant matrix A can now be written as

$$\begin{aligned} A &= \begin{bmatrix} P & X \\ 0 & Q \end{bmatrix} \begin{bmatrix} \Lambda & 0 \\ 0 & J \end{bmatrix} \begin{bmatrix} P & X \\ 0 & Q \end{bmatrix}^{-1} \\ &= \begin{bmatrix} P\Lambda P^{-1} & -P\Lambda P^{-1}XQ^{-1} + XJQ^{-1} \\ 0 & QJQ^{-1} \end{bmatrix} \end{aligned} \tag{142}$$

The unknown X is solution of

$$\begin{aligned} -A_{11}XQ^{-1} + XJQ^{-1} &= A_{12} \\ A_{11}X - XJ &= -A_{12}Q \end{aligned} \tag{143}$$

expanding:

$$(P^{-1}A_{11}P)(P^{-1}X) - P^{-1}XJ = -P^{-1}A_{12}Q \tag{144}$$

with $X = PY$ and $PF = -A_{12}Q$ we get

$$\Lambda Y - YJ = F \tag{145}$$

or

$$\begin{bmatrix} f_{11} & f_{12} & \cdots & f_{1m} \\ f_{21} & f_{22} & \cdots & f_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ f_{n1} & f_{n2} & \cdots & f_{nm} \end{bmatrix}$$

$$= \begin{bmatrix} (\lambda_1 - j_{11})y_{11} & (\lambda_1 - j_{22})y_{12} - y_{11}j_{12} & \cdots & (\lambda_1 - j_{mm})y_{1m} - y_{1,m-1}j_{m-1,m} \\ (\lambda_2 - j_{21})y_{21} & (\lambda_2 - j_{22})y_{22} - y_{21}j_{12} & \cdots & (\lambda_2 - j_{mm})y_{2m} - y_{2,m-1}j_{m-1,m} \\ \vdots & \vdots & \ddots & \vdots \\ (\lambda_n - j_{n1})y_{n1} & (\lambda_n - j_{nn})y_{n2} - y_{n1}j_{12} & \cdots & (\lambda_n - j_{mm})y_{nm} - y_{n,m-1}j_{m-1,m} \end{bmatrix}$$

This equation can be solved with the following algorithm:

Do $l = 1, n$

$y_{l,1} = \frac{f_{l,1}}{\lambda_l - j_{11}}$

Do $k = 2, m$

$y_{l,k} = \frac{f_{l,k} + y_{l,k-1}j_{k-1\,k}}{\lambda_l - j_{kk}}$

End do

End do

Aircraft Dynamics

The part of the plant matrix that contains the dynamics of the aircraft is not ill conditioned. The eigendecomposition does therefore not present any major problems. Various available algorithms can be used.

Turbulence Part

The dynamics of the Dryden turbulence model are given by the following state matrix.

$$\begin{bmatrix} -\frac{U_0}{l_0} & 0 & 0 & 0 & 0 \\ 0 & 0 & (\frac{U_0}{2l_0})^2 & 0 & 0 \\ 0 & -1 & -\frac{U_0}{l_0} & 0 & 0 \\ 0 & 0 & 0 & 0 & (\frac{U_0}{2l_0})^2 \\ 0 & 0 & 0 & -1 & -\frac{U_0}{l_0} \end{bmatrix} \tag{146}$$

let us define

$$a = \frac{U_0}{2l_0} \tag{147}$$

Equation (146) is composed of one diagonal element $-2a$ and two diagonal blocks having the form

$$A = \begin{bmatrix} 0 & a^2 \\ -1 & -2a \end{bmatrix} \tag{148}$$

We are interested in finding the Jordan form of the matrix A. The only eigenvalue of the matrix A is $\lambda = \lambda_1 = \lambda_2 = -a$. The Jordan matrix J can therefore be written as:

$$J = \begin{bmatrix} -\lambda & 1 \\ 0 & -\lambda \end{bmatrix} = Q^{-1}AQ \tag{149}$$

let

$$Q = \begin{bmatrix} q_{11} & q_{12} \\ q_{21} & q_{22} \end{bmatrix} \quad \text{and} \quad Q^{-1} = \frac{1}{q_{11}q_{22} - q_{12}q_{21}} \begin{bmatrix} q_{22} & -q_{12} \\ -q_{21} & q_{11} \end{bmatrix} \tag{150}$$

Equation (149) is then

$$\begin{bmatrix} -a & 1 \\ 0 & -a \end{bmatrix} = \frac{1}{\det Q} \begin{bmatrix} q_{12}q_{11} + q_{21}a^2q_{22} + 2q_{21}q_{12}a & q_{12}^2 + a^2q_{22}^2 + 2q_{22}q_{12}a \\ -q_{11}^2 - a^2q_{21}^2 + 2q_{11}q_{21}a & -q_{11}q_{12} - q_{21}a^2q_{22} - 2q_{11}q_{22}a \end{bmatrix} \tag{151}$$

The solution to these four equations is given by:

$$\begin{aligned} q_{11} &= -aq_{21} \\ q_{22} &= -(q_{21} + q_{12})/a \end{aligned} \tag{152}$$

for any values of q_{12} and q_{21}. One possible solution of the Jordan decomposition of the turbulence matrix (eq. 146) is thus given by:

$$J = \begin{bmatrix} -\frac{U_0}{2l_0} & 0 & 0 & 0 & 0 \\ 0 & -\frac{U_0}{2l_0} & 1 & 0 & 0 \\ 0 & 0 & -\frac{U_0}{2l_0} & 0 & 0 \\ 0 & 0 & 0 & -\frac{U_0}{2l_0} & 1 \\ 0 & 0 & 0 & 0 & -\frac{U_0}{2l_0} \end{bmatrix} \quad \text{and} \quad Q = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & -\frac{U_0}{2l_0} & 1 & 0 & 0 \\ 0 & 1 & -\frac{4l_0}{U_0} & 0 & 0 \\ 0 & 0 & 0 & -\frac{U_0}{2l_0} & 0 \\ 0 & 0 & 0 & 1 & -\frac{U_0}{2l_0} \end{bmatrix}$$

CHAPTER VIII

MEAN SQUARE VALUES

Actually, not only the performance index $J(u_{opt})$, which is the weighted sum of the mean square values of the states and the inputs, is of interest to us, but also the respective mean square values themselves. They show us how well the pilot succeeds in maintaining the airplane in the steady state maneuver ($\mathrm{E}\{x_i^2\}$) and how big the control effort is ($\mathrm{E}\{u_j^2\}$). They also allow us to see which axis is the most difficult to control.

Let us consider the optimal closed loop system without time delays (equation 95). (The system with time delays can be handled similarly.)

$$\dot{x} = (A - BL)x + Ew \tag{153}$$

In chapter VII, we have shown that the expected mean square value of the state vector x, is a solution of the matrix differential equation

$$\dot{\Sigma} = (A - BL)\Sigma + \Sigma(A - BL)^{\mathrm{T}} + EWE^{\mathrm{T}} \tag{154}$$

where $\Sigma = \mathrm{E}\{xx^{\mathrm{T}}\}$.

In the steady state case this equation reduces to the algebraic Lyapunov equation:

$$(A - BL)\Sigma + \Sigma(A - BL)^{\mathrm{T}} + EWE^{\mathrm{T}} = 0 \tag{155}$$

which can be solved with the algorithm described by Bartels and Stewart [12].

Knowing the expected mean square value of x, the expected value of the input is given by (see VII):

$$\mathrm{E}\{uu^{\mathrm{T}}\} = L\Sigma L^{\mathrm{T}} \tag{156}$$

The mean square values of the state variables and the input variables are thus the diagonal elements of the matrices Σ and $L\Sigma L^{\mathrm{T}}$, respectively.

CHAPTER IX

WEIGHTING MATRICES

The most important drawback of the optimal pilot model is that it is not clear what are good choices for the weighting matrices Q and R. But, since in this paper we are interested in comparing the value of the performance index for various maneuvers and not to get a good correlation with the reality, the 'correct' selection of these matrices is not that much of a concern. The weighting matrices are assumed to be diagonal and the terms are choosen using Bryson's rule:

$$q_{ii} = \frac{1}{x^2_{imax}} \quad r_{jj} = \frac{1}{u^2_{jmax}} \tag{157}$$

Let the maximum control surface deflections be:

elevator : $\delta_{e_{max}} = \mp 25$ deg (from data)
aileron : $\delta_{a_{max}} = \mp 10$ deg (estimated)
rudder : $\delta_{r_{max}} = \mp 20$ deg (estimated)

These are related to the maximum control stick and rudder pedal deflections:

$$\delta_{e_s} = \frac{\delta_e}{K_e} \tag{158}$$

$$\delta_{a_s} = \frac{\delta_a}{K_a} \tag{159}$$

$$\delta_{r_s} = \frac{\delta_r}{K_r} \tag{160}$$

The input weighting matrix is then:

$$R = \begin{bmatrix} \frac{K_e}{\delta^2_{emax}} & 0 & 0 \\ 0 & \frac{K_a}{\delta^2_{amax}} & 0 \\ 0 & 0 & \frac{K_r}{\delta^2_{rmax}} \end{bmatrix} \tag{161}$$

The gearing constants for the modeled aircraft are:

$$K_e \cong 2.47 \text{ [deg /inch]} \tag{162}$$

$$K_a = 2.07 \text{ [deg /inch]} \tag{163}$$

$$K_r = 6.756 \text{ [deg /inch]} \tag{164}$$

The actuator time constant τ is typically 0.05 seconds. The weighting matrix R has now the following numerical values:

$$R = \begin{pmatrix} 9.76\ 10^{-3} & 0 & 0 \\ 0 & 4.28\ 10^{-2} & 0 \\ 0 & 0 & 0.1411 \end{pmatrix} \tag{165}$$

The output states are defined to be the sideslip angle β, the bank angle ϕ and the pitch angle θ. Assuming the following maximal deviations

$$\beta_{max} = 4.0 \text{ deg} \tag{166}$$

$$\phi_{max} = 5.0 \text{ deg} \tag{167}$$

$$\theta_{max} = 3.0 \text{ deg} \tag{168}$$

the output weighting matrix Q becomes:

$$Q = \begin{pmatrix} 205 & 0 & 0 \\ 0 & 131 & 0 \\ 0 & 0 & 365 \end{pmatrix} \tag{169}$$

CHAPTER X

PROGRAM AND RESULTS

Program Description

To calculate the value of the performance index for the OCM, a Fortran computer program has been developed. Whenever possible, routines of the battle-damaged aircraft simulation program by Computational Engineering, Inc. [10] are used. Figure 4 shows the flow chart of the program. The user is prompted to input the altitude, the Mach-number and the percentage of fuel remaining. Next, the desired maneuver with the necessary parameters is entered through a menu driven input subroutine. The maneuvers implemented are:

1. horizontal unaccelerated flight.

2. steady state turn with either the bank angle, the g-load or the rate of turn specified.

3. symmetrical pullup with the acceleration and the pitch angle given.

From the steady state values, the program determines the stability derivatives and other pertinent parameters by interpolation in an aircraft data table (see Appendix A). The system matrices A, B, C and E are then set up. The optimum control problem is solved next. The Riccati-equation is solved using a Schur-decomposition method [11]. The value of the performance index for the system with time delays and the system without can be calculated. The expected mean square values are found by solving the Lyapunov-equation [12]. The output given by the program is:

1. the value of the performance index

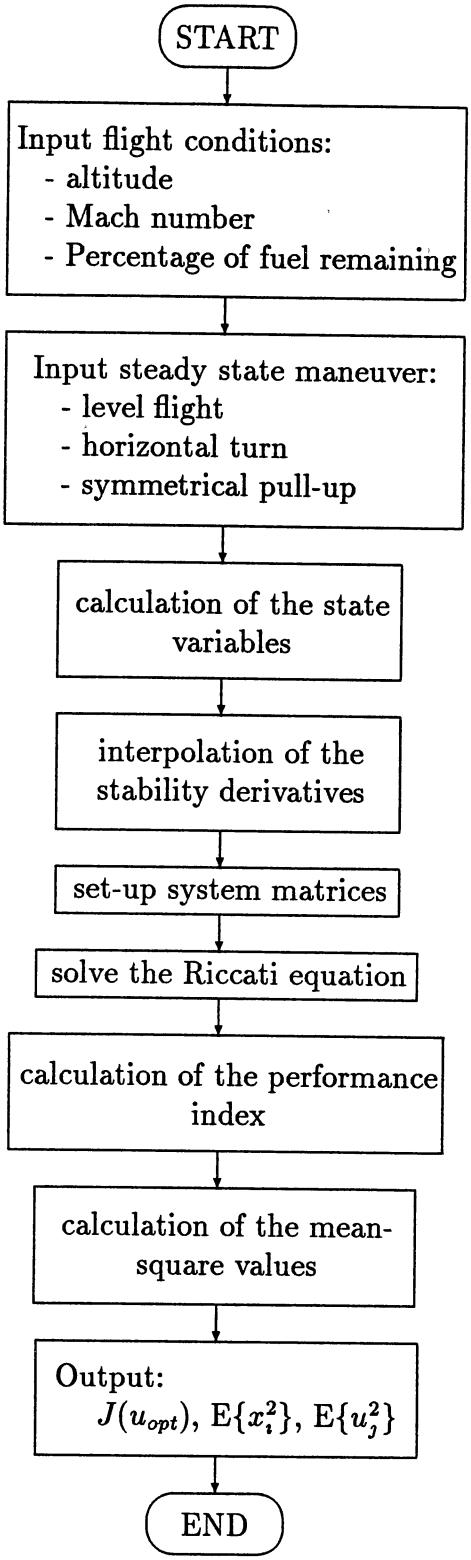


Figure 4. Flowchart of the program

2. the mean square values of the control inputs (elevator, aileron, rudder) and the mean square values of the sideslip angle, the bank angle and the pitch angle.

In Appendix C, a compilation of all the subroutines used is made. The listings of the subroutines which had to be modified or specifically written for this program are also given. Other subroutines used come from the software packets Linpack [13] and Eispack [14]. The Riccati- and Lyapunov-equation solver were provided by A.J. Laub.

Results

To simplify the calculations and to eliminate one of the parameters, the time delay τ is initially set to zero. The output weighting matrix is as described in Chapter IX.

The performance index is calculated for the maneuvers:

1. steady state turn with variable bank angle.

2. symmetrical pull-up with constant acceleration at various pitch angles.

3. symmetrical pull-up at zero pitch angle with increasing acceleration.

Steady State Turn

Table I shows the mean square values of the sideslip angle β, the bank angle ϕ and the pitch angle θ, the elevator stick deflections, the aileron stick deflection and the rudder pedal input and the calculated performance index as a function of the bank angle Φ_0 of the steady state turn.

The root mean square values of the variables at 0 and at 60 degrees is given in table II. For comparison, the maximum assumed deflections used for the calculation of the weighting matrices are also shown.

TABLE I

PERFORMANCE INDEX AND MEAN SQUARE VALUES FOR STEADY STATE TURN

Φ_0	$E\{\beta^2\}$	$E\{\phi^2\}$	$E\{\theta^2\}$	$E\{\delta_{e_s}^2\}$	$E\{\delta_{a_s}^2\}$	$E\{\delta_{r_s}^2\}$	$J(u_{opt})$
	$\times 10^5$	$\times 10^7$	$\times 10^6$		$\times 10^2$	$\times 10^2$	
deg	rad^2	rad^2	rad^2	in^2	in^2	in^2	-
0	1.084	2.904	3.578	37.38	5.295	1.142	0.3722
5	1.084	3.167	3.588	37.38	5.295	1.142	0.3723
10	1.084	3.949	3.619	37.39	5.295	1.142	0.3724
15	1.084	5.227	3.672	37.41	5.296	1.141	0.3726
20	1.084	6.963	3.752	37.43	5.297	1.141	0.3728
25	1.084	9.100	3.865	37.45	5.298	1.141	0.3732
30	1.084	11.56	4.018	37.48	5.300	1.141	0.3735
35	1.084	11.34	3.611	36.31	5.300	1.141	0.3620
40	1.084	13.42	3.841	36.38	5.302	1.141	0.3628
45	1.084	15.47	4.162	36.47	5.303	1.141	0.3638
50	1.084	14.81	4.139	35.93	5.304	1.141	0.3584
55	1.084	16.13	4.725	36.20	5.306	1.141	0.3614
60	1.084	15.23	5.175	36.31	5.307	1.142	0.3626

TABLE II

ROOT MEAN SQUARE VALUES OF VARIABLES
AND MAXIMUM ALLOWED DEFLECTIONS
FOR HORIZONTAL STEADY
STATE TURN

Φ_0 deg	β deg	ϕ deg	θ deg	δ_{e_s} inch	δ_{a_s} inch	δ_{r_s} inch	$J(u_{opt})$ -
0	0.19	0.031	0.108	6.11	0.23	0.11	0.3722
60	0.19	0.071	0.130	6.00	0.23	0.11	0.3626
max:	4.0	5.0	3.0	10.1	4.8	3.0	

The changes in the variances of the pitch angle are not steady. This is partly due to the values of the stability derivatives, the longitudinal moment coefficient C_{m_α} in particular (see Fig. 5). One can observe that the sideslip angle and the rudder input variances are essentially independent of the bank angle. The elevator control input decreases slightly (about 7 %) between 0 and 60 degrees bank angle. An increase in the variance of the pitch angle is therefore expected and can actually be observed (20 %). The aileron input remains almost constant, but the bank angle variance increases by a factor of about 2.3. Even though the workload in the lateral axis increases, the variance of the bank angle increases.

In summary, for increasing steady state bank angles the variance of the pitch and the bank angle increases. For the same amount of control work, the output variance gets worse. The airplane is more difficult to fly. But, this increase in difficulty is not reflected in the value of the performance index, which actually decreases. Because of the mean square value and the respective weight it has in

the calculation of the performance index, the influence of the elevator input is the most important. And, therefore, since the variance of the elevator decreases, the value of the performance index gets smaller too.

Symmetrical Pull-up

The quasi-steady symmetrical pull-up with a constant pitch angle and varying normal acceleration is considered next (Table III). The sideslip angle and the bank angle remain essentially constant. So do the rudder and the aileron input. This can be expected since this maneuver is in the vertical plane only. But, both the pitch and the elevator variance are decreasing for higher g-loads. This goes against our intuition.

Essentially the same observation can be made for the symmetrical constant acceleration pull-up at various pitch angles (Table IV).

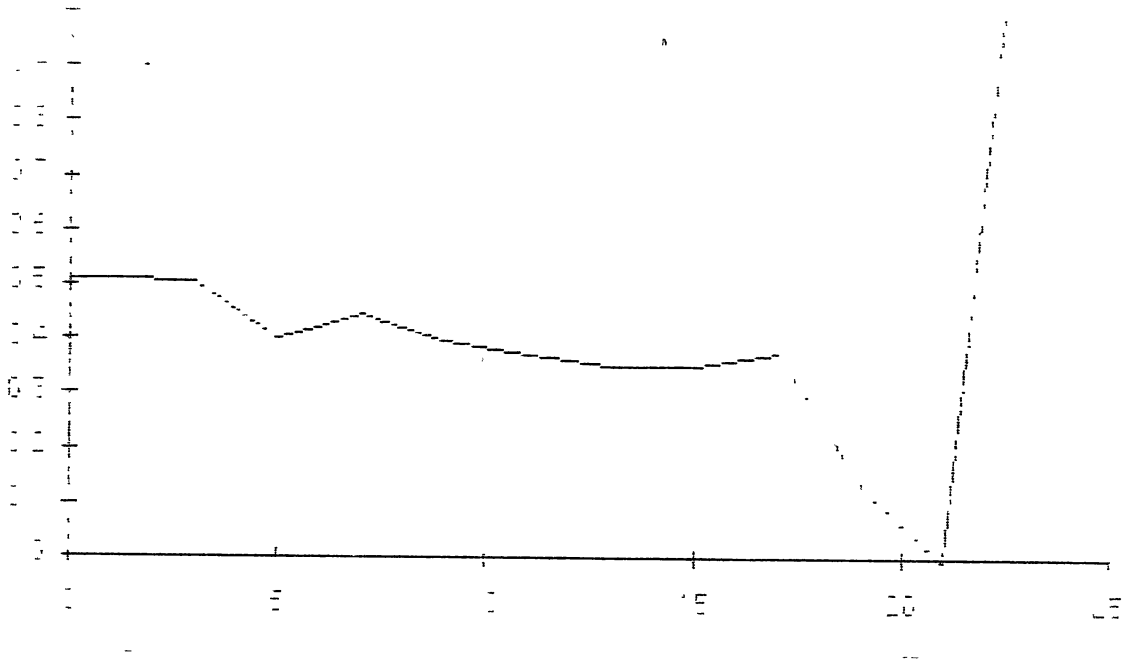


Figure 5. Longitudinal moment derivative C_{m_α} as a function of the steady state angle of attack.

TABLE III

PERFORMANCE INDEX AND MEAN SQUARE VALUES
FOR QUASI-STEADY SYMMETRICAL PULLUP
AT PITCH ANGLE $\Theta_0 = 0$ deg

n	$E\{\beta^2\}$	$E\{\phi^2\}$	$E\{\theta^2\}$	$E\{\delta_{e_s}^2\}$	$E\{\delta_{a_s}^2\}$	$E\{\delta_{r_s}^2\}$	$J(u_{opt})$
	$\times 10^5$	$\times 10^7$	$\times 10^6$		$\times 10^2$	$\times 10^2$	
g	rad^2	rad^2	rad^2	in^2	in^2	in^2	-
1.0	1.084	2.904	3.578	37.38	5.295	1.142	0.3722
1.2	1.084	2.902	3.790	37.50	5.295	1.141	0.3735
1.4	1.085	2.900	3.407	36.51	5.295	1.141	0.3737
1.6	1.085	2.897	3.202	36.07	5.295	1.141	0.3794
1.8	1.085	2.895	3.288	36.45	5.295	1.141	0.3631
2.0	1.085	2.892	3.137	37.48	5.295	1.141	0.3634
2.2	1.085	2.890	3.221	37.16	5.295	1.141	0.3700
2.4	1.085	2.887	3.266	39.91	5.295	1.141	0.3773

TABLE IV

PERFORMANCE INDEX AND MEAN SQUARE VALUES
FOR QUASI-STEADY SYMMETRICAL PULL-UP
AT CONSTANT G-LOAD $N = 2$

Θ_0	$E\{\beta^2\}$	$E\{\phi^2\}$	$E\{\theta^2\}$	$E\{\delta_{e_s}^2\}$	$E\{\delta_{a_s}^2\}$	$E\{\delta_{r_s}^2\}$	$J(u_{opt})$
	$\times 10^5$	$\times 10^7$	$\times 10^6$		$\times 10^2$	$\times 10^2$	
deg	rad^2	rad^2	rad^2	in^2	in^2	in^2	-
0	1.085	2.892	3.137	36.48	5.295	1.141	0.3634
20	1.085	2.896	3.077	36.44	5.295	1.141	0.3629
40	1.085	2.907	2.938	36.30	5.296	1.141	0.3615
60	1.085	2.924	2.707	36.07	5.297	1.140	0.3592
80	1.085	2.946	2.402	35.76	5.298	1.140	0.3560
100	1.085	2.972	2.072	35.41	5.299	1.140	0.3525
120	1.085	2.997	1.782	35.06	5.300	1.139	0.3490
140	1.085	3.019	1.572	34.77	5.301	1.139	0.3461
160	1.085	3.033	1.454	34.58	5.301	1.139	0.3442
180	1.085	3.038	1.423	34.51	5.302	1.138	0.3435
200	1.085	3.033	1.476	34.58	5.301	1.139	0.3441

CHAPTER XI

DISCUSSION

Airplane dynamics

The variation of the performance index $J(u)$ is not as expected. A maneuver that intuitively should be more difficult to fly yields smaller values of the performance index. This could be caused by the changes in the plant matrix A due to the steady state values, the linearization is performed about. The system matrix changes both because of the stability derivatives which can be functions of the steady state variables, and the steady state variables themselves. We can get some insight in what happens by studying how the poles of the plant matrix change as a function of the maneuver. Table V and VI give the eigenvalues of the system for the steady turn maneuver and the symmetrical pull-up at zero degree pitch. Figures 6 and 7 show a plot of the root loci of the modes with the lowest damping ratios for the two maneuvers, respectively.

We see that the trends in both cases are exactly opposite. Whereas the damping ratios tend to increase for the pull-up, the damping ratios decrease for the turn. Actually, at some point, the plant even gets unstable.

This explains why the mean square values increase with greater bank angles for the turn and decrease with higher g-load factors for the pull-up. In the first case the dynamics become worse whereas they get better in the second case.

2^{nd} order model

But this still doesn't explain the values of the performance index. In an attempt to get more insight into the optimal control theory for a plant with

TABLE V

EIGENVALUES OF THE PLANT MATRIX AS
A FUNCTION OF THE STEADY STATE
BANK ANGLE IN A TURN

Φ_0	eigenvalues				
0	-0.952 ± 2.617j	-0.006 ± 0.032j	-0.079 ± 1.770j	-2.600	-0.030
10	-0.952 ± 2.617j	-0.005 ± 0.031j	-0.079 ± 1.771j	-2.600	-0.031
20	-0.952 ± 2.617j	-0.003 ± 0.029j	-0.078 ± 1.773j	-2.602	-0.035
30	-0.952 ± 2.617j	-0.000 ± 0.026j	-0.077 ± 1.779j	-2.607	-0.040
40	-0.953 ± 2.470j	-0.003 ± 0.023j	-0.074 ± 1.787j	-2.614	-0.047
50	-0.955 ± 2.373j	-0.005 ± 0.018j	-0.068 ± 1.802j	-2.629	-0.055
60	-0.960 ± 2.304j	-0.005 ± 0.014j	-0.055 ± 1.829j	-2.675	-0.072

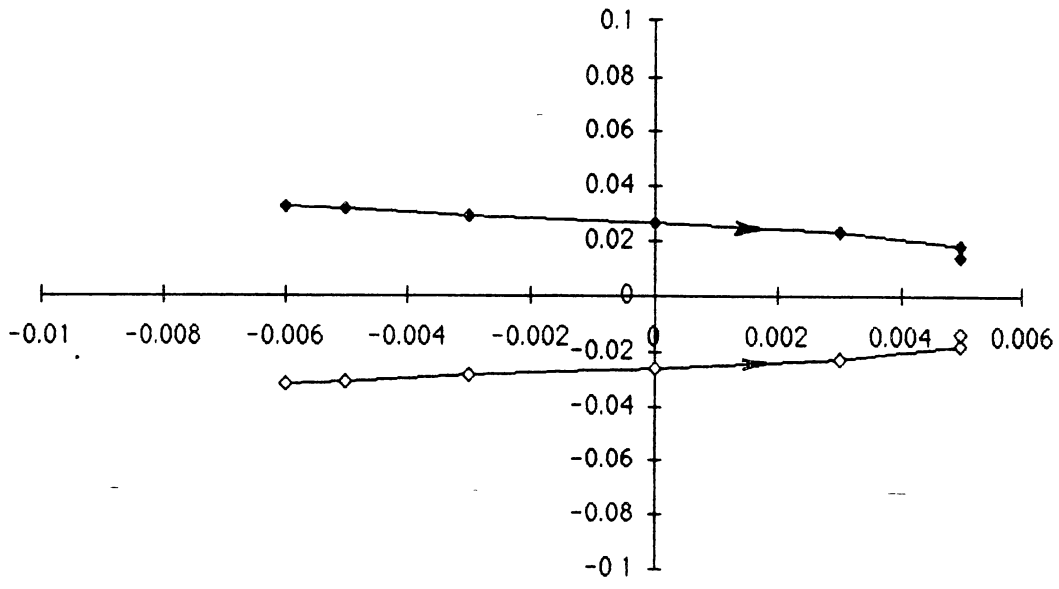


Figure 6. Root-locus of the eigenmode with the lowest damping for steady state turn (0 - 60 deg bank).

TABLE VI

EIGENVALUES OF THE PLANT MATRIX AS A FUNCTION OF THE NORMAL ACCELERATION IN A SYMMETRICAL PULL-UP

g-load	eigenvalues				
1	-0.952 ± 2.617j	-0.006 ± 0.032j	-0.079 ± 1.770j	-2.600	-0.030
1.4	-0.954 ± 2.469j	-0.010 ± 0.052j	-0.095 ± 1.742j	-2.568	-0.031
1.8	-0.958 ± 2.373j	-0.017 ± 0.065j	-0.112 ± 1.713j	-2.535	-0.032
2.0	-0.961 ± 2.303j	-0.023 ± 0.070j	-0.121 ± 1.699j	-2.519	-0.032
2.2	-0.964 ± 2.313j	-0.029 ± 0.074j	-0.130 ± 1.684j	-2.501	-0.033
2.6	-0.971 ± 2.403j	-0.044 ± 0.078j	-0.148 ± 1.653j	-2.466	-0.034
3.0	-0.980 ± 1.341j	-0.066 ± 0.065j	-0.166 ± 1.621j	-2.429	-0.036

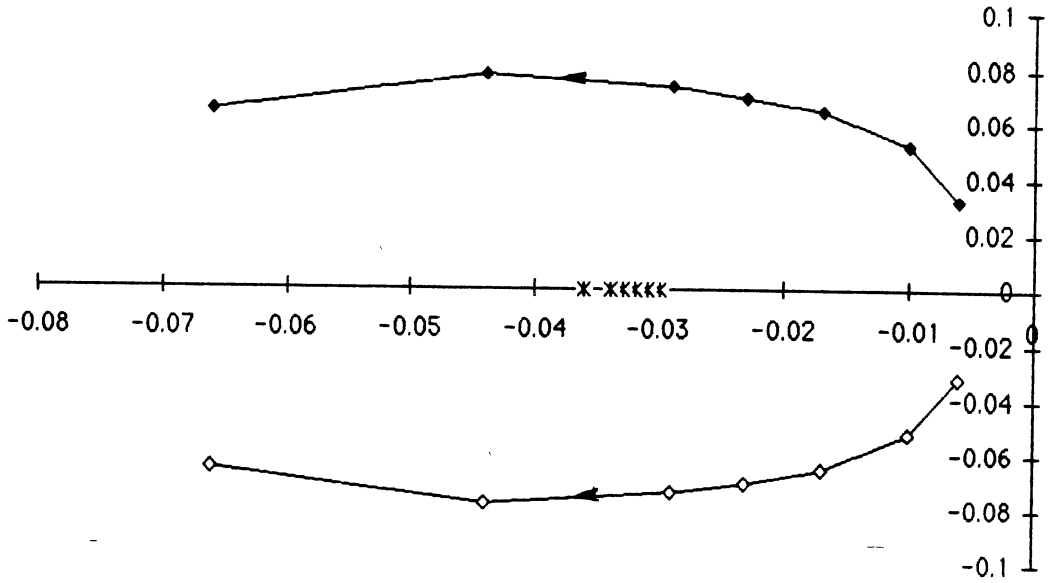


Figure 7. Root-locus of the eigenmode with the lowest damping for steady state pull-up.

changing dynamics, a second order system is considered.

$$\dot{x} = \begin{bmatrix} -2\zeta\omega_0 & -\omega_0^2 \\ 1 & 0 \end{bmatrix} x + \begin{bmatrix} 1 \\ 1 \end{bmatrix} u \tag{170}$$

The input u is assumed to be unit white noise ($\mathrm{E}\{uu^T\} = I\delta(t-\tau)$). The optimum control problem is solved for various combinations of ω_0 and ζ. The constant weighting matrices used are:

$$Q = \begin{bmatrix} 10 & 0 \\ 0 & 5 \end{bmatrix} \quad R = 1 \tag{171}$$

In table VII, the values of the performance index calculated using equation (102) are given. It can be seen that, for decreasing values of the damping ratios (less favorable handling qualities), the performance index decreases (for $\mathrm{Re}\{\lambda\} < 0$). The same is true for the control input. The variances of the output do not have any regular pattern. Even in this simple problem the variables do not behave as expected. This seems to be inherent to the optimal control theory used.

Further comments and future work

Addition of a time delay has the effect to increase the value of the performance index as can be seen in equation (122). Since even without the time delay the performance index does not reflect the difficulty of the maneuver flown, the influence of the time delay was not further investigated.

The main reason why the performance index does not necessarily increase with more difficult dynamics can be attributed to the fact that the mean square values of the output and the input both change, and often in opposing directions. If one can somehow force the mean square values of either the input or the output to remain independent of the maneuvers, then the change in the other mean square values and the value of the performance index should reflect the difficulty of the maneuver. To do this, the weighting matrices Q and R must be functions of the maneuver. At this point, there is no method to calculate the values of the elements of these matrices as a function of the plant matrix A, to guarantee constant mean

TABLE VII

PERFORMANCE INDEX AND MEAN-SQUARE VALUES FOR 2^{ND} ORDER SYSTEM

		eigenvalue	mean-square values			
ζ	ω_0	λ	x_1	x_2	u	$J(u_{opt})$
1.0	0.2	-0.20 ± 0.9798j	0.1292	0.1514	2.2716	4.3203
0.9	0.2	-0.18 ± 0.8818j	0.1249	0.1565	2.2586	4.2903
0.8	0.2	-0.16 ± 0.7838j	0.1211	0.1610	2.2562	4.2728
0.7	0.2	-0.14 ± 0.6859j	0.1180	0.1647	2.2646	4.2677
0.6	0.2	-0.12 ± 0.5879j	0.1322	0.1333	2.2450	4.2336
0.5	0.2	-0.10 ± 0.4899j	0.1275	0.1397	2.1876	4.1609
0.4	0.2	-0.08 ± 0.3919j	0.1235	0.1452	2.1561	4.1177
0.3	0.2	-0.06 ± 0.2939j	0.1299	0.1306	2.1297	4.0818
0.2	0.2	-0.04 ± 0.1960j	0.1258	0.1372	2.0651	4.0087
0.1	0.2	-0.02 ± 0.0980j	0.1273	0.1331	2.0061	3.9445
0.0	0.2	0.00 ± 0.0000j	0.1425	0.1083	2.6513	4.6179
-0.1	0.2	0.02 ± 0.0980j	0.1291	0.1294	2.0535	3.9916
-0.2	0.2	0.04 ± 0.1960j	0.1293	0.1299	2.1621	4.1043
-0.3	0.2	0.06 ± 0.2939j	0.1339	0.1223	2.3161	4.2662

square values. They have to be found by trial and error, which is not a very practical way since a lot of iterations may be required. Also, it may be necessary to include non-zero off-diagonal terms.

CHAPTER XII

CONCLUSION

The optimal control model was used to simulate the pilot controlling an airplane for various steady state maneuvers. The plant matrix is obtained by linearizing the dynamics about the steady state condition and the optimal control model is solved. It was found that the value of the performance index $J(u)$ does not reflect the difficulty of the maneuver flown. Also, the mean square values of the control inputs and the output states do not change as expected. By using a simple second order system, it has been shown that this behavior is inherent to the optimal control theory if constant weighting matrices are used.

One of the main problems in using the optimal control theory is the choice of the weighting matrices. In this work they were assumed to be independent of the maneuver. But this assumption makes it difficult to compare two maneuvers because all the mean square values can change. Further research is necessary on how to choose the weighting matrices as a function of the plant dynamics so that the mean square values of either the inputs or the outputs remain constant.

BIBLIOGRAPHY

1. Harper, R. P. Jr. and Cooper, G. E.;" Handling Qualities and Pilot Evaluation", *Journal of Guidance, Control, and Dynamics*, Vol. 9, Sept./Oct. 1986.

2. Thompson, P. M. and McRuer, D. T.;"A Comparison of the Human Optimal Control and Crossover Models", *AIAA Paper No. 88-4183.*

3. McLean, D.; "Automatic Flight Control Systems", Prentice Hall (1990).

4. Kleinman, D.L., Baron, S., Levison, W. H.;"A Control Theoretic Approach to Manned Vehicle Systems Analysis", *IEEE Transactions on Automatic Control*, Vol. AC-16, NO. 6, December 1971.

5. Etkin, B.;"Dynamics of Flight — Stability and Control", second edition, John Wiley & Sons (1982).

6. Swaim, R. L.;"An Alternative LQG Pilot Model Derivation", unpublished.

7. Roskam, J.; "Airplane Flight Dynamics and Automatic Flight Controls. Part 1", Roskam Aviation and Engineering Corporation (1982).

8. Roskam, J.; "Methods for Estimating Stability and Control Derivatives of Conventional Subsonic Aircraft", The University of Kansas, Lawrence, Kansas, (1973).

9. Kleinman, D. L.;"Optimal Control of Linear Systems with Time-Delay and Observation Noise", *IEEE Transactions on Automatic Control*, , December 1969.

10. Anonymous;"Dynamics and Control of Battle-Damaged Aircraft", CE43-90-12/mas 1222, Computational Engineering, Inc., P.O. Box 1595, Dahlgren, Virginia 22448, February 1990.

11. Laub, A. J.;"A Schur Method for Solving Algebraic Riccati Equations", *IEEE Transactions on Automatic Control*, Vol. AC-24, 1979.

12. Bartels, R. H. and Stewart, G. W.;"Solution of the Matrix Equation $AX + X^TB = C$", Algorithm 432, Comm. ACM, 15, 1972.

13. Dongarra, J. J.,Bunch, J.R., Moler, G.W. Stewart; "Linpack Users' Guide", SIAM, Philadelphia (1979).

14. Smith, B. T., Boyle J. M., Dongarra, J.J., Garbow, B. S., Ikebe, Y., Klema, V. C.,Moler, C. B.;"Matrx Eigensysem Routines - EISPACK Guide", Second Edition, Springer Verlag, 1976.

15. Golub, J. H. and van Loan C. F.;"Matrix Computations", The Johns Hopkins University Press, Baltimore, Maryland (1983).

APPENDICES

APPENDIX A

AIRCRAFT DATA TABLE

On the following pages, the aircraft data tables which are used in this study are given. They are for a fighter type aircraft with a variable sweep wing. At the reference condition, the wing is fully swept.

0.00	2.84	9 78	16.72	23.66	30.60	37 54	44.48	51.42	58 36	65 30	72.24	79.18	86.12	93 06	100.00
24273	24273	24966	25659.	26352	27045	27738	28431.	29124	29817	30510	31203	31896.	32589	33282	33975
35 75	35 75	35 60	35.45	34 65	34 65	35 60	35 80	35 05	34 43	34 45	34 76	35 75	36 75	36 50	36 25
0.00	0.00	0 00	0.00	0.00	0.00	0 00	0.00	0 00	0 00	0.00	0.00	0.00	0 00	0 00	0.00
-.669	-.669	- 670	-.675	- 673	-.670	-.650	-.715	- 706	- 702	-.705	-.715	- 725	-.695	- 680	-.675
12100	12100	12120.	12200	12200.	12300	12400.	12700.	13000.	13200	14700.	17500.	19200.	19400.	19600.	19800.
86200	86200	87000	88000	89000.	89800	90600.	91200.	91600	92200.	93000.	93600.	94800.	96800.	98800.	100600.
94600.	94600.	95400	96400	97400.	98200	99200.	100000.	100600	101400	103800	107200	110200	112200	114400	116200
16.3	25 5	367 7													
19.85	0 00	0 00	0 00	180 10	0 00	0 00	0.00	180 10							
		← blank →													
2		3	4	.5	.6	.7	.8	9	1.0	1.1	1 2	1.3	1 4		
200		400.	600.	800	1000.	1200.	1400	1600.							
-4.		2	4.	6	8	10.	12	14	16.	18.	20.	22.	24		
-25.	-24	-22	-20.	-18.	-16	-14.	-12.	-10.	10.	12.	14	16	18	20.	22.	24.	25
.00	05	10	15	20	.25	.30	.40	50	60	.70	.80	.90	1.00				
		← blank →													
-.0275	- 0275	- 0275	- 0275	- 0275	-.0146	-.0125	-.0152	-.0668	-.0228	-.0307	-.0374	-.0433			
-.0139	- 0139	- 0139	-.0139	- 0139	-.0127	-.0109	-.0220	-.0647	-.0206	-.0142	-.0086	-.0198			
-.0113	- 0113	-.0113	-.0113	-.0113	-.0121	-.0145	-.0237	- 0627	-.0197	-.0107	-.0003	-.0022			
-.0135	- 0135	- 0135	-.0135	- 0135	-.0146	-.0203	-.0234	-.0614	-.0193	-.0129	-.0079	- 0060			
-.0156	-.0156	-.0156	- 0156	-.0156	-.0171	-.0261	-.0231	- 0601	-.0189	- 0151	-.0155	-.0098			
- 0179	- 0179	-.0179	-.0179	-.0179	- 0196	- 0319	- 0228	-.0588	-.0185	-.0173	- 0231	-.0136			
-.0200	- 0200	- 0200	-.0200	- 0200	-.0221	-.0377	- 0225	- 0575	-.0181	-.0195	-.0307	-.0174			
- 0223	-.0223	- 0223	-.0223	-.0223	-.0246	-.0435	-.0222	- 0562	-.0177	-.0217	-.0383	-.0212			
0507	0507	0507	.0507	.0507	.0531	.0602	.0652	0565	.0569	.0599	.0628	.0657			
.0494	0494	0494	.0494	0494	0526	.0562	.0612	.0538	.0516	.0492	.0468	.0444			
.0479	0479	0479	0479	.0479	.0512	.0542	.0592	0522	.0492	.0452	.0412	0372			
.0462	0462	0462	.0462	.0462	0502	.0532	0582	.0512	.0482	.0442	.0402	.0362			
.0446	0446	0446	.0446	0446	.0491	.0521	.0571	.0501	.0471	.0431	.0391	.0351			
0430	0430	.0430	.0430	.0430	0480	.0510	.0560	0490	.0460	.0420	.0380	.0340			
.0414	0414	0414	.0414	.0414	0469	.0499	.0549	0479	.0449	.0409	.0369	.0329			
.0398	.0398	0398	0398	0398	.0458	.0488	.0538	.0468	.0438	.0398	0358	.0318			
-.0082	- 0082	-.0082	- 0102	-.0103	-.0120	-.0114	-.0116	- 0118	-.0118	-.0118	-.0097	-.0076			
		← blank →													
.0150	0175	0196	.0220	.0244	.0268	.0295	0325	.0304	.0290	.0280	.0270	.0262			
.0170	.0170	0170	.0170	.0170	0170	.0170	.0170	0370	.0395	.0395	0372	.0350			
.0242	.0242	.0242	.0242	.0242	0242	0242	.0242	0482	.0540	.0555	.0550	.0540			
.0321	0321	.0321	.0321	.0321	.0321	.0321	.0321	0608	.0690	.0723	.0722	.0725			
.0421	0421	0421	.0421	0421	.0421	.0430	0438	0750	.0845	.0890	.0903	.0915			
.0550	0550	0550	.0550	.0550	.0550	.0565	0583	.0910	.1006	1058	1082	.1105			
.0690	.0690	0690	.0690	0690	0691	.0723	0749	.1080	.1166	.1228	1263	.1300			
0840	0840	0840	0840	0840	.0868	.0905	0928	.1268	.1334	.1405	.1447	.1495			
.1188	1188	1188	.1188	.1188	.1230	.1283	.1319	.1649	.1710	.1788	.1842	.1900			
.1560	1560	1560	.1560	.1560	1620	.1680	.1722	.2062	.2135	2217	.2280	.2341			
1961	.1961	.1961	.1961	1961	.2028	.2090	2148	.2510	.2584	.2667	2761	.2833			
.2391	2391	2391	2391	.2391	.2460	.2528	.2591	.2983	.3055	.3155	.3262	.3360			
.2840	.2840	.2840	.2840	2840	.2920	.2890	.3045	.3473	.3540	.3655	.3787	.3887			
.3300	3300	3300	.3300	3300	.3390	.3465	.3527	3968	.4036	.4163	.4325	4414			
.3760	3760	1760	3760	.3760	.3860	4040	4009	.4463	.4532	.4671	.4863	.4941			
		← blank →													

PERCENT FUEL REMAINING
WEIGHT AS FUNCTION OF FUEL LOADING (lbs)
X C G LOCATION AS PERCENT OF M A C FOR EACH FUEL LOADING
Y C G LOCATION AS FUNCTION OF FUEL LOADING (ft)
Z C G LOCATION AS FUNCTION OF FUEL LOADING (ft)
I_{xx} AS FUNCTION OF FUEL LOADING (slug ft²)
I_{yy} AS FUNCTION OF FUEL LOADING (slug ft²)
I_{zz} AS FUNCTION OF FUEL LOADING (slug ft²)
AERODYNAMIC REFERENCE GEOMETRY – CHORD SPAN AREA (ft ft ft²)
ENGINE ROTOR INERTIA TENSOR (slug ft²)

MACH NUMBER ARRAY
DYNAMIC PRESSURE ARRAY (lbs/ft²)
ANGLE-OF-ATTACK ARRAY (deg)
ELEVATOR DEFLECTION ARRAY (deg)
C_L^2 ARRAY

C_{L_δ} vs MACH NUMBER ——→
vs
DYNAMIC PRESSURE

C_{L_δ} vs MACH NUMBER - - -→
vs
DYNAMIC PRESSURE

$C_{L_{\delta_e}}$ vs MACH NUMBER

$C_{D_{SB}}$ (SPEED BRAKE)
$C_{D_{SBM}}$ vs MACH NUMBER
vs
C_L^2

- 01515	- 01515	-.01515	-.01515	- 01515	- 01515	- 01515	-.01515	- 01505	- 01505	- 01502	- 01510	- 01520	C_{Y_β} vs MACH NUMBER
00210	00210	00210	00210	00210	00210	00210	00157	00130	.00120	00095	00080	00075	$C_{Y_{\delta_r}}$ vs MACH NUMBER
	← blank →												
- 00240	00240	00240	00240	- 00240	- 00265	-.00305	- 00225	- 00148	- 00148	- 00148	- 00148	- 00148	C_{l_β} vs MACH NUMBER
0042	0042	0042	0042	.0042	.0042	.0042	0038	0038	.0038	0038	0038	0038	C_{l_r} vs MACH NUMBER
- 0119	-.0119	-.0119	-.0119	- 0119	- 0119	-.0119	-.0105	-.0105	- 0113	- 0113	-.0112	-.0110	C_{l_p} vs MACH NUMBER
00025	00025	.00025	.00025	.00025	00025	.00025	.00021	00018	00016	00013	00012	00010	$C_{l_{\delta_r}}$ vs MACH NUMBER
00106	00106	00106	00106	00106	00106	00106	00122	00128	.00132	00134	00118	00102	$C_{l_{\delta_a}}$ vs MACH NUMBER
00040	00040	00040	00040	00040	00040	.00056	00056	.00044	00030	00022	00016	00002	$C_{l_{\delta_{sp}}}$ vs MACH NUMBER
	← blank →												
- 0097	0097	0097	0097	- 0097	-.0086	- 0080	- 0070	.0125	0030	-.0020	0022	- 0025	C_{m_0} vs MACH NUMBER
0360	0360	0360	.0360	0360	0386	.0412	0540	.0768	0736	0672	0616	0560	C_m DUE TO α vs MACH NUMBER
- 0180	- 0180	- 0180	- 0180	- 0180	-.0193	- 0206	-.0207	-.0384	- 0368	- 0336	- 0308	-.0208	vs
- 0357	-.0357	-.0357	- 0357	-.0357	- 0385	-.0412	- 0540	-.0768	- 0736	- 0672	- 0614	- 0556	ANGLE-OF-ATTACK (α)
- 0525	-.0525	- 0525	- 0525	- 0525	- 0570	- 0615	- 0805	- 1148	- 1100	-.1004	- 0915	- 0826	
- 0680	- 0680	- 0680	-.0680	-.0680	- 0736	- 0791	- 1050	- 1520	- 1450	- 1320	- 1201	- 1082	
- 0819	-.0819	-.0819	-.0819	- 0819	-.0889	- 0958	-.1205	-.1834	- 1767	-.1624	- 1471	- 1318	NOTE NON-DERIVATIVE FORM
- 0948	- 0948	- 0948	-.0948	- 0948	-.1032	-.1116	- 1210	-.2050	-.2027	- 1924	- 1731	-.1538	
- 1070	- 1070	- 1070	-.1070	-.1070	- 1166	- 1262	-.1200	-.2240	- 2287	- 2224	- 1991	- 1758	
- 1193	-.1193	-.1193	-.1193	- 1193	-.1289	-.1385	- 1360	-.2430	-.2547	-.2524	- 2251	-.1978	
- 1325	-.1325	- 1325	-.1325	- 1325	- 1413	-.1500	-.1510	- 2620	-.2800	-.2820	- 2510	- 2200	
- 1372	- 1372	-.1372	- 1372	-.1372	- 1456	-.1540	-.1550	- 2810	-.3050	-.3100	- 2750	- 2400	
-.1370	-.1370	-.1370	-.1370	- 1370	-.1455	- 1540	- 1550	-.3000	-.3250	- 3290	-.2960	- 2630	
- 1880	- 1880	-.1880	-.1880	- 1880	-.1680	-.1480	- 1520	-.3180	- 3400	- 3450	- 3150	-.2850	
0084	0084	.0084	.0084	0084	0084	0088	.0092	.0093	0087	0080	0072	0065	$C_{m_{\delta_e}}$ vs MACH NUMBER
-.06632	06632	06632	- 06632	- 06632	- 06632	-.07679	- 09599	-.05760	- 05585	- 05411	- 05236	-.05236	C_{m_q} vs MACH NUMBER
	← — blank →												
.00270	00238	00210	00190	00170	00150	.00090	.00149	00240	.00260	00290	.00275	.00250	C_{n_β} vs MACH NUMBER
- 0101	-.0101	-.0101	-.0101	-.0101	-.0105	- 0108	-.0119	- 0131	-.0147	-.0138	-.0134	- 0126	C_{n_r} vs MACH NUMBER
- 0021	-.0021	-.0021	-.0021	-.0021	-.0021	-.0026	-.0030	- 0031	-.0026	-.0019	-.0019	- 0019	C_{n_p} vs MACH NUMBER
-.00133	- 00130	-.00128	-.00126	- 00124	-.00120	-.00118	- 00100	-.00080	-.00065	- 00050	- 00044	-.00040	$C_{n_{\delta_r}}$ vs MACH NUMBER
.00002	.00006	.00012	.00014	00022	00025	00028	.00032	00040	.00045	00029	00010	00000	$C_{n_{\delta_{sp}}}$ vs MACH NUMBER
00080	00080	00080	00080	.00080	00080	.00080	.00080	00090	00100	00104	00096	00085	$C_{n_{\delta_a}}$ vs MACH NUMBER
	← — blank →												
88[illegible]	882	882	882	882	.885	.887	.878	.927	.960	.970	.970	970	K_{δ_e} vs MACH NUMBER →
882	.882	.882	.882	882	.885	887	878	.927	960	970	970	.970	vs
.912	.912	.912	.912	912	.909	906	898	939	967	.976	976	976	ELEVATOR DEFLECTION
.922	922	.922	.922	922	.925	928	.918	.961	.973	.980	980	.980	
.941	.941	941	.941	941	.944	.946	938	.963	980	.985	985	985	
.960	.960	.960	.960	.960	964	967	957	977	988	.990	990	990	
.979	.979	.979	.979	.979	982	.985	997	988	994	995	995	995	
1.000	1.000	1.000	1.000	1.000	1.000	1.000	997	1.000	1 000	1.000	1 000	1 000	
1 000	1 000	1.000	1.000	1 000	1 000	1.000	1.000	1 000	1.000	1 000	1 000	1.000	
1 000	1.000	1.000	1.000	1 000	1.000	1.000	1 000	1 000	1.000	1 000	1 000	1.000	
1.000	1.000	1 000	1 000	1.000	1.000	1.000	.997	1 000	1.000	1 000	1 000	1.000	
.979	.979	979	.979	979	982	.985	.997	988	.994	995	.995	995	
960	960	960	.960	960	.964	.967	957	977	988	.990	.990	990	
.941	.941	.941	.941	.941	.944	.946	.938	.963	.980	985	985	.985	
.922	.922	.922	.922	922	925	928	918	961	.973	.980	980	980	
.912	.912	.912	.912	912	909	.906	.898	.939	.967	.976	976	.976	
.882	.882	.882	882	882	.885	.887	.878	.927	.960	.970	.970	970	
.882	882	882	882	.882	.885	.887	.878	.927	.960	970	970	970	
	← — — blank →												

(ELEVATOR EFFECTIVENESS TERM)

1.000	1.000	940	.900	875	850	825	800	SUBSONIC
1 000	1 000	920	870	835	800	765	730	SUPERSONIC — AEROELASTIC CORRECTION TERM TO C_m DUE TO α vs DYNAMIC PRESSURE
1 000	1 000	985	970	955	950	940	935	SUBSONIC
1 000	1 000	970	.940	920	900	880	860	SUPERSONIC — AEROELASTIC CORRECTION TERM TO $C_{m_{\delta_e}}$ vs DYNAMIC PRESSURE
1.000	1 000	940	880	.830	760	700	650	AEROELASTIC CORRECTION TERM TO $C_{Y_{\delta_r}}$ AND $C_{Y_{\delta_r}}$ vs DYNAMIC PRESSURE
←	– blank –	→						
0	10000	20000	30000	40000	50000	60000		ALTITUDE ARRAY (ft)
←	blank	→						
810	1100	1650	2400	3210.	4440	5930.	8220	11030. 14600 18320. 22250. 26300 — ENGINE RAMDRAG vs MACH NUMBER →
580	740	1190.	1700.	2270	3120	4150	5730	7670 9800 12190 15000 18000. — vs
400	600	840	1200	1580.	2140.	2830.	3880.	5180. 6700. 8380. 10100. 12050. — ALTITUDE
270	390	570	750.	1060	1430.	1870.	2560.	3400 4350 5400 6400 7840 — (FLIGHT IDLE POWER SETTING - lbs)
170	250	370	510	680.	910.	1190.	1620.	2140. 2740. 3380. 4000. 4820
110.	160	230	300.	420	560.	730	1000	1330 1680. 2100 2500 3050
81	120	170	210.	280	320	450	600	800 1120. 1500 1860 2200
←	blank	→						
1687	2530	3503	4581.	5823.	7520.	9339	11852.	14942. 17528. 20596. 24250. 28605 — ENGINE RAMDRAG vs MACH NUMBER →
1210.	1815	2511	3275	4169.	5232	6494.	8218	10504 12373. 14568. 17176. 20292 — vs
869.	1303.	1796.	2267	2869.	3605	4487	5655	7075 8346. 10037 11838. 13992 — ALTITUDE
615	922	1330	1587.	1906.	2391	2975	3733.	4664 5548. 6574 7754 9352
391	586	841	1009.	1206.	1514.	1880.	2355	2939. 3515. 4186 4939 5840 — (MILITARY POWER SETTING lbs)
241	362	520	624	746.	936.	1163	1457.	1817 2174. 2589 3054. 3611
149	224	322	386	461	579.	719	901	1124. 1344 1601 1889 2233
←	blank	→						
1687	2530	3503	4581	5823	7520.	9339.	11852	14942 17528. 20596 24250. 28605 — ENGINE RAMDRAG vs MACH NUMBER →
1210	1815	2511	3275.	4169.	5232.	6494	8218.	10504 12373. 14568. 17176 20292 — vs
869	1303	1796	2267	2869.	3605.	4487	5655	7075. 8346. 10037 11838. 13992 — ALTITUDE
615	922	1330	1587	1906.	2391.	2975	3733	4664 5548. 6574. 7754. 9352.
391	586	841.	1009	1206	1514.	1880	2355	2939 3515. 4186. 4939. 5840. — (FULL AFTERBURNER POWER SETTING - lbs)
241	362	520	624	746	936	1163.	1457	1817 2174. 2589. 3054. 3611
149	224	322	386	461.	579	719	901	1124. 1344. 1601 1889 2233.
←	blank	→						
2250	2500	2780	3280	3770	4290.	5000	5770	6770 8150. 9830. 11880. 14380 — GROSS THRUST vs MACH NUMBER →
1360.	1550	1800	2050	2400	2720.	3150	3740	4440 5390. 6540. 7690. 9680 — vs
690.	800	1000	1190	1490	1650.	1900.	2260.	2730 3360. 4140. 5080. 6230 — ALTITUDE
260	350.	440	570.	700	880	1090.	1320	1600. 1950. 2450. 3030. 3770
-80	-10	40.	120	210.	320	440	600.	790 1020. 1280. 1610. 2070 — (FLIGHT IDLE POWER SETTING - lbs)
-330	-300	-260	-210	-150	-90.	-10.	80.	200 340. 510. 710. 1000.
-460	-430	-415	-400	-340	-300.	-220	-140	-50. 50. 180 330 550.
←	– blank	→						
16000	17100	18600	19900	21146	22650.	24400.	26400	28600. 31100. 33800. 36800. 40100. — GROSS THRUST vs MACH NUMBER →
12460.	13259	14074.	14875	15843	16977.	18253.	19737	21398 23236. 25261. 27460. 29850 — vs
8900.	9396	10016	10655	11399.	12328.	13376.	14583	15903. 17377. 19007. 20777. 22687. — ALTITUDE
5970.	6330	6770	7255	7826	8542	9356.	10293.	11346. 12498. 13769. 15171. 16703.
3803.	4024	4305	4634	5026	5520.	6081	6730.	7463. 8274. 9171. 10148. 11205. — (MILITARY POWER SETTING lbs)
2800.	2900	3100	3300.	3600.	4000.	4450	5000	5600. 6250. 7000. 7850. 8750.
1700	1800	2000	2250	2600	2900	3250	3700	4250 4750. 5300. 5950 6700.
←	– blank	→						

25800	26800	28500.	30750	33300	35950	38900	42100	46200	50000	53800	57600	61400	GROSS THRUST AT MACH NUMBER
18900.	19850	21000.	22505.	23935	25659	27735.	30450.	35110	37743	40597	42368	44139	
14000	14500	15000	15900	16950	18200	19850	21850	24250	27050	30000	32950	36000	ALTITUDE
9300	9750	10200.	10926	11754	12787	14137	15563	17301	19399	21536	23818	26541	
5600	5900	6270	6820.	7450	8200	8980	9960	11117	12486	14042	15778	17557	
3300	3500.	3800	4200	4640	5150	5680	6280	6980	7800	8700	9700	10700	
1650	1800	2070	2400	2700	3050	3400.	3850	4300	4800	5300	5900	6600	

← - blank ⟶

-6 05	-4 00	-5 50	-5 00	-4 50	-4 00	-3 50	-3 00	-2 50	-2 00	-1 50	-1 00	-0 50	0 00	0 50	LONGITUDINAL STICK POSITION (inches)
1 00	1 50	2.00	2.50	3.00	3 50	4 00	4 50	5 00	5 50						
-8 50	-8 30	-5 70	-3 55	-1.80	-0 30	0 85	1 80	2 50	3.10	3 60	3 95	4.40	5 00	5 70	COMMANDED ELEVATOR POSITION (degrees)
6 70	7 80	9 20	10.70	12 45	14 50	16 80	19 40	22 20	24 00						

0 00	0 50	1.00	2.00	3.00	4 00	4.50	4.80	4.84	LATERAL STICK POSITION (inches)
-0 00	-00	-4 40	-16 20	-30 10	-40 40	-43 50	-45.00	-45.00	COMMANDED SPOILER POSITION (degrees)

← blank ⟶

2.070	6 756	1 420	31 000	-14 500	3 700	-3 700	CAIL, CRUD, DO, MAXSTB MINSTB, MAXPED MINPED
-40	40	-50	-99	40	40.	50.	99. MNRATE(1-4) MXRATE(1-4)

← — blank ⟶

20 0	40 0	30 0	0 8	3.5	2 0	DDEMAX DDRMAX, DDAMAX, DECMAX, DRCMAX, DACMAX
0.	1228	-0.35	-0.07	-0 210	-0.035	QDSAS(1-2), AKQDMP(1-2), AKPDMP(1-2)
0 50	-0 20	-0 18	-0 25	-0 13		KRDAMP, KBDAMP, KPSI, KTHETA KPHI

} SEE DEFINITIONS BELOW

CAIL – AILERON GEARING CONSTANT (deg/inch)
CRUD – RUDDER GEARING CONSTANT (deg/inch)
DO – ZERO OFFSET FOR SPOILER STICK POSITION (inches)
MAXSTB MINSTB – MAXIMUM AND MINIMUM STABILATOR POSITIONS (degrees)
MAXPED MINPED – MAXIMUM AND MINIMUM RUDDER PEDAL POSITIONS (inches)
MNRATE, MAXRATE – MAXIMUM AND MINIMUM DEFLECTION RATES FOR ELEVATOR, AILERON, RUDDER AND SPOILER DEFLECTIONS (deg/sec)
DDEMAX, DDRMAX DDAMAX – MAXIMUM STABILITY AUGMENTATION SYSTEM AUTHORITY LIMITS FOR ELEVATOR, RUDDER AND AILERON DEFLECTION RATES (deg/sec)
DECMAX, DRCMAX, DACMAX – MAXIMUM STABILITY AUGMENTATION SYSTEM AUTHORITY LIMITS FOR ELEVATOR RUDDER AND AILFRON DFFLECTIONS (deg)
QDSAS – DYNAMIC PRESSURE ARRAY FOR DAMPING GAINS (lbs/ft)
AKQDMP – PITCH DAMPING GAIN ARRAY vs DYNAMIC PRESSURE (sec)
AKPDMP – ROLL DAMPING GAIN ARRAY vs DYNAMIC PRESSURE (sec)
KRDMP – YAW DAMPING GAIN (sec)
KBDMP – SIDESLIP DAMPING GAIN (sec)
KPSI – HEADING ANGLE ATTITUDE HOLD GAIN (sec)
KTHETA – ELEVATION ANGLE ATTITUDE HOLD GAIN (sec)
KPHI – BANK ANGLE ATTITUDE HOLD GAIN (sec)

APPENDIX B

NUMERICAL EXAMPLE OF STATE MATRICES

The numerical values of the plant matrix A, the control matrix B and the noise matrix E are given for the horizontal unaccelerated flight condition. The Mach-number for this example is M=0.5, the altitude 10000 ft and the percentage of fuel left is 70 %.

A =

Columns 1 thru 8

-0.0129	0.0000	-0.0015	0.0000	0.0000	0.0000	0.0000	0.0000
0.0000	-0.1298	0.0000	0.0000	0.0000	-1.0000	0.0000	0.0000
-0.0196	0.0000	-0.4408	0.0000	1.0000	0.0000	0.0374	0.0000
0.0000	-18.5206	0.0000	-2.2231	0.0000	0.7846	0.0000	8.1799
0.0070	0.0000	-5.0208	0.0000	-1.1200	0.0000	6.2262	0.0000
0.0000	1.9976	0.0000	-0.0535	0.0000	-0.2571	0.0000	0.8411
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	-20.0000	0.0000
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	-20.0000
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	1.0000	0.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	1.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000

Columns 9 thru 16

0.0000	0.0000	-0.0611	0.0000	0.0000	0.0000	0.0000	0.0000
0.0180	0.0611	0.0000	0.0000	0.0002	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0008	0.0000
1.9292	0.0000	0.0000	0.0000	0.0352	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	-0.1469	0.0000
-1.3248	0.0000	0.0000	0.0000	-0.0040	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
-20.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	-0.3009	0.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	0.0000	0.0226	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	-1.0000	-0.3009	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0226
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	-1.0000	-0.3009

E =

0.0000	0.0000	0.0000
0.0000	0.0000	0.0000
0.0000	0.0000	0.0000
0.0000	0.0000	0.0000
0.0000	0.0000	0.0000
0.0000	0.0000	0.0000
0.0000	0.0000	0.0000
0.0000	0.0000	0.0000
0.0000	0.0000	0.0000
0.0000	0.0000	0.0000
0.0000	0.0000	0.0000
2.6263	0.0000	0.0000
0.0000	2.2744	0.0000
0.0000	-14.1147	0.0000
0.0000	0.0000	2.2744
0.0000	0.0000	-14.1147

B =

0.0000	0.0000	0.0000
0.0000	0.0000	0.0000
0.0000	0.0000	0.0000
0.0000	0.0000	0.0000
0.0000	0.0000	0.0000
0.0000	0.0000	0.0000
0.4780	0.0000	0.0000
0.0000	0.7220	0.0000
0.0000	0.0000	2.3580
0.0000	0.0000	0.0000
0.0000	0.0000	0.0000
0.0000	0.0000	0.0000
0.0000	0.0000	0.0000
0.0000	0.0000	0.0000
0.0000	0.0000	0.0000
0.0000	0.0000	0.0000

APPENDIX C

PROGRAM AND SUBROUTINES

Program PILMOD calls:

- AEROX: reads in aerodynamic data table
- SETUP† : input of initial conditions
- INIT: interpolates mass and inertia data
- ATMOS: calculates aircraft velocity as a function of altitude and Mach-number
- PLTMOD†: pilot model

PLTMOD calls:

- SSMAN† : defines the steady state maneuver flown
- PMADER† : calculates the stability derivatives by interpolation in the data tables.
- PIMAM† : sets up the plant matrix A.
- RICSOL† : driver routine to solve the Riccati-equation.
- GOFTAU : Calculates the transition matrix e^{At}.
- INTEGRAL† : Solves the integral $\int_0^T e^{At} Q e^{A^T t}\, dt$
- LYAP† : driver routine to solve the Lyapunov equation.

†listings of these subroutines are given on the next pages

RICSOL calls:

- RICCND : RICPACK subroutine to solve the Riccati-equation.

INTEGRAL calls:

- EIGV : calcualtes the eigenvalues and eigenvectors of a square matrix.
- CGECO : LINPACK subroutine used to solve system of equations.
- CGESL : LINPACK subroutine used to solve system of equations.

LYAP calls:

- LYPCND : calculates the solution of the Lyapunov equation.

Other subroutines used:

- TRNPOS : calculates the transpose of a matrix.
- MATMUL : calculates the product of two real matrices.
- CMATMUL : calculates the product of two complex matrices.
- TRACE : calculates the trace of a square matrix.

```
      PROGRAM PILMOD

      INTEGER  NN , NR , NM , NL , NLPNR
      CHARACTER*1 CHAR
      PARAMETER (NN=16,NR=3,NM=3,NL=8,NLPNR=11)

C        ***************************************************************
C        *          THE COMMON BLOCK /DRYDEN/ CONTAINS THE             *
C        *    THE CONSTANTS USED BY THE DRYDEN GUST VELOCITY MODEL     *
C        ***************************************************************
      REAL  L0 , SIGMA0
      COMMON /DRYDEN/  L0, SIGMA0
C        ***************************************************************
C        *    THE COMMON BLOCK /AERO1/ CONTAINS THE AERODYNAMIC        *
C        *    COEFFICIENT TABLES THAT ARE READ IN FROM THE DATA        *
C        *    FILE.  THE AERODYNAMIC COEFFICIENTS ARE THEN             *
C        *    INTERPOLATED IN THESE TABLES TO OBTAIN INSTANTANEOUS     *
C        *    VALUES.                                                  *
C        ***************************************************************
      REAL
     1   M(13) , QD(8) , A(13) , E(18) , CL2(14) , ACLFT0(13,8) ,
     2   ACLFTA(13,8) , ACLFTE(13) , ACDSB(13) , ACDTRM(13,14) ,
     3   ACYB(13) , ACYDR(13) , ACLB(13) , ACLR(13) , ACLP(13) ,
     4   ACLDR(13) , ACLDA(13) , ACLDSP(13) , ACM0(13) , ACMA(13,13) ,
     5   ACMDE(13) , ACMQ(13) , ACNB(13) , ACNR(13) , ACNP(13) ,
     6   ACNDR(13) , ACNDSP(13) , ACNDA(13) , AKDE(13,18) , NMASUB(8) ,
     7   NMASUP(8) , NMESUB(8) , NMESUP(8) , ANDR(8)
      COMMON /AERO1/
     1   M,QD,A,E,CL2,ACLFT0,ACLFTA,ACLFTE,ACDSB,ACDTRM,ACYB,ACYDR,
     2   ACLB,ACLR,ACLP,ACLDR,ACLDA,ACLDSP,ACM0,ACMA,ACMDE,ACMQ,
     3   ACNB,ACNR,ACNP,ACNDR,ACNDSP,ACNDA,AKDE,NMASUB,NMASUP,
     4   NMESUB,NMESUP,ANDR

C        ***************************************************************
C        *     THE COMMON BLOCK /MASCAR/ CONTAINS THE MASS RELATED     *
C        *  PROPERTIES OF THE AIRCRAFT. INCLUDED ARE THE ENGINE        *
C        *  ROTOR (IROTOR) AND AIRCRAFT INERTIA TENSOR (IAC)           *
C        *  MATRICIES, AS WELL AS THE AIRCRAFT MASS AND CENTER OF      *
C        *  MASS LOCATION.                                             *
C        ***************************************************************
      REAL  IROTOR(3,3) , IAC(3,3) , ACMASS , RCM(3)
      COMMON /MASCAR/
     1      IROTOR , IAC , ACMASS , RCM

C         ***************************************************************
C         *     THE COMMON BLOCK /CNTRLS/ CONTAINS THE DEFLECTIONS      *
C         *  OF THE AIRCRAFT CONTROL SURFACES.  INCLUDED IN THIS        *
C         *  COMMON BLOCK ARE THE RUDDER DEFLECTION (DRUD), THE         *
C         *  SPOILER DEFLECTION (DSPOIL), THE SYMMETRICAL               *
C         *  DEFLECTION OF THE HORIZONTAL TAIL (DELEV), THE             *
C         *  ASSYMMETRIC DEFLECTION OF THE HORIZONTAL TAIL (DAIL),      *
C         *  AND THE SPEED BRAKE DEFLECTION (DSB).                      *
C         ***************************************************************
       REAL  DRUD , DSPOIL , DELEV , DAIL , DSB
       COMMON /CNTRLS/
     1             DRUD , DSPOIL , DELEV , DAIL , DSB

C        ***************************************************************
C        *     THE COMMON BLOCK /ACGEOM/ CONTAINS AIRCRAFT GEOMETRY    *
```

```
C        *  INFORMATION.  INCLUDED IN THIS COMMON BLOCK ARE THE      *
C        *  AIRCRAFT'S WINGSPAN (SPAN), THE AIRCRAFT'S WING AREA     *
C        *  (AREA), AND THE MEAN AERODYNAMIC CHORD (CHORD).          *
C        ************************************************************
      REAL  SPAN , AREA , CHORD
      COMMON /ACGEOM/
     1            SPAN , AREA , CHORD


C     ************************ PROGRAM VARIABLES *********************
      REAL PI,G0,MACH,ALT,FUEL,GAMMA,LOAD,PHI,RHO,
     1     ASOS,SGMAN,VC,QBAR,FOPT(NR,NN),GOPT(NR,NM),KLIMIT,
     2     ALPHA, THETA
      INTEGER SED,THROTL
      LOGICAL ERRFLG
C        ******************** FORMAT STATEMENTS ***********************

 1010 FORMAT(5X,I15,/,5X,F5.2,F10.0,F10.0,9X,I1)

C        ******************** START EXECUTABLE CODE *******************
 15    CONTINUE
C        ************************************************************
C        *         INITIALIZE COUNTERS AND PHYSICAL CONSTANTS       *
C        ************************************************************
      PI = 3.1415927
      G0=32.174049
      ERRFLG = .FALSE.
C        ************************************************************
C        *        READ AERODYNAMIC DATA FROM DATAFILE
C        ************************************************************
      CALL AEROX
C        ************************************************************
C        *           SET INITIAL CONDITIONS FOR SIMULATION          *
C        ************************************************************
      OPEN(UNIT=7,FILE='ACSET.DAT',STATUS='OLD')
      READ(7,1010)SED,MACH,ALT,FUEL,THROTL
      CLOSE(UNIT=7,STATUS='KEEP')

      CALL SETUP
     B          (MACH , ALT , FUEL , THROTL ,
     Y          GAMMA , LOAD , PHI )


C        ************************************************************
C        *   CALL INITIALIZATION SUBROUTINE TO READ IN THE DATABASE  *
C        ************************************************************
      CALL INIT
     G         ( FUEL )
C        ************************************************************
C        *               CALCULATE AIRCRAFT VELOCITIES              *
C        ************************************************************
      CALL ATMOS
     G        ( ALT
     Y        , RHO , ASOS , SGMAV )

      VC = MACH*ASOS
      QBAR = .5*RHO*VC*VC
```

```
C        ****************************************************************
C        *            CALL PLTMOD SUBROUTINE TO CALCULATE               *
C        *               THE PERFORMANCE INDEX                          *
C        ****************************************************************

      CALL PLTMOD
     G    ( MACH , ACMASS , IAC , QBAR ,
     G      VC , 0.0 , 0.0 , 0 , 1 ,
     Y      FOPT , GOPT , KLIMIT , ERRFLG )
      IF (ERRFLG) THEN
C        ***** ERROR ABORT *****
        write(6,*)' MACS, ERROR IN PLTMOD:'
        GOTO 200
      END IF

 200  CONTINUE
      WRITE(*,'(1X,''ENTER 1 FOR ANOTHER RUN'')')
      READ*,ONE
      IF (ONE .EQ.1)goto 15
      STOP
      END
```

```
      SUBROUTINE SETUP
     G               (MACH , ALT , FUEL , THROTL ,
     Y               GAMMA , LOAD , PHI )
C    ****************************************************************
C    *                                                              *
C    *    CALLING SEQUENCE                                          *
C    *        SUBROUTINE SETUP                                      *
C    *                                                              *
C    *    DESCRIPTION  :  SETUP                                     *
C    *                    -----                                     *
C    *        SUBROUTINE SETUP READS THE INITIAL FLIGHT             *
C    *    CONDITIONS, DAMAGE OPTION, DATA SET CHOICE, AND PILOT     *
C    *    OPTIONS FOR THE SIMULATION FROM A DATA FILE(ACSET.DAT).   *
C    *    THE USER IS THEN ALLOWED TO CHANGE THESE CONDITIONS       *
C    *    THROUGH A SERIES OF MENUS FOR THE INDIVIDUAL RUN.         *
C    *    AT THE END OF THE SIMULATION, THE USER IS GIVEN THE       *
C    *    CHOICE TO SAVE THESE FLIGHT CONDITIONS AS NEW DEFAULT     *
C    *    VALUES FOR FUTURE RUNS.                                   *
C    *                                                              *
C    *    RESTRICTIONS                                              *
C    *        (NONE)                                                *
C    *                                                              *
C    *    COMMUNICATION                                             *
C    *        ARGUMENT LIST                                         *
C    *            GIVEN                                             *
C    *                ENDTME (R) = SIMULATION LENGTH (sec)          *
C    *            BOTH                                              *
C    *                MACH   (R) = DESIRED MACH NUMBER AT START     *
C    *                             OF SIMULATION                    *
C    *                ALT    (R) = DESIRED ALTITUDE AT START        *
C    *                             OF SIMULATION (ft)               *
C    *                FUEL   (R) = DESIRED PERCENT OF REMAINING     *
C    *                             FUEL                             *
C    *                THROTL (I) = THROTTLE SETTING                 *
C    *                                  1 = FLIGHT IDLE             *
C    *                                  2 = MILITARY                *
C    *                                  3 = AFTER BURNER            *
C    *            YIELDED                                           *
C    *                GAMMA  (R) = CLIMB/DIVE ANGLE AT START OF     *
C    *                             SIMULATION (deg)                 *
C    *                LOAD   (R) = DESIRED LOAD FACTOR FOR PULL     *
C    *                             MANEUVER OR TURN                 *
C    *                PHI    (R) = BANK ANGLE AT START OF           *
C    *                             SIMULATION (deg)                 *
C    *                                                              *
C    *    LOCAL VARIABLES                                           *
C    *        ANSWER (A) = ANSWER TO YES/NO QUESTIONS               *
C    *        OPTION (I) = NUMBER OF FLIGHT CONDITION TO BE         *
C    *                     CHANGED BY USER                          *
C    *        SASOPT (A) = CHARACTER VARIABLE " ON" OR "OFF"        *
C    *                     USED TO TELL USER OF THE STABILITY       *
C    *                     AUGMENTATION SYSTEM STATUS               *
C    *        ROLOPT (A) = CHARACTER VARIABLE " ON" OR "OFF"        *
C    *                     USED TO TELL USER OF ROLL                *
C    *                     AUGMENTATION CHANNEL STATUS              *
C    *        CHAR1  (A) = CHARACTER VARIABLE USED IN MENUS         *
C    *        CHAR2  (A) = CHARACTER VARIABLE USED IN MENUS         *
C    *        CHAR3  (A) = CHARACTER VARIABLE USED IN MENUS         *
C    *        CHAR4  (A) = CHARACTER VARIABLE USED IN MENUS         *
C    *        CHAR5  (A) = CHARACTER VARIABLE USED IN MENUS         *
```

```
C       *                                                              *
C       *
C       *        (NONE)                                                *
C       *                                                              *
C       *    ERROR CHECKS                                              *
C       *        (NONE)                                                *
C       *                                                              *
C       *    ASSOCIATED SUBPROGRAMS                                    *
C       *        CALLED BY                                             *
C       *            MACS    (AIRCRAFT SIMULATION)                    *
C       *        CALLS TO                                              *
C       *            SETTRJ                                            *
C       *                                                              *
C       *    REFERENCES                                                *
C       *        (NONE)                                                *
C       *                                                              *
C       *    LANGUAGE                                                  *
C       *        STANDARD FORTRAN 77                                   *
C       ****************************************************************

C       ********** ARGUMENT VARIABLES **********
      REAL  ENDTME , MACH , ALT , FUEL , CMDS(7,4) , GAMMA , LOAD ,
     1      PHI , DMGTME
      INTEGER  THROTL , NUMCMD , DAMAGE , ADATA , ADATA2 , PDATA ,
     1         PDATA2
      LOGICAL  SASON , ROLLON

C       ********** LOCAL VARIABLES **********
      INTEGER  OPTION
      CHARACTER*1  ANSWER
      CHARACTER*3  SASOPT , ROLOPT
      CHARACTER*38  CHAR1 , CHAR6
      CHARACTER*16  CHAR2 , CHAR3 , CHAR4 , CHAR5
      REAL  PI

C       ********** DATA STATEMENTS **********
      DATA  PI /3.1415927/


C
C       ********** START EXECUTABLE CODE **********
C

C       ************************************************************
C       *        WRITE FLIGHT CONDITION MENU TO THE SCREEN         *
C       ************************************************************
   10 WRITE(6,20)MACH,ALT,FUEL
   20 FORMAT(10(/),15X,25HCURRENT FLIGHT CONDITIONS,//,8X,
     1      15HMACH NUMBER  = ,29X,F4.2/,8X,
     2      20HALTITUDE IN FEET  = ,22X,F6.0,/,8X,
     3      29HPERCENT OF FUEL REMAINING  = ,15X,F4.0,//)

C       ************************************************************
C       *      GIVE CHOICE TO CHANGE CURRENT FLIGHT CONDITIONS      *
C       ************************************************************
      WRITE(6,21)
   21 FORMAT(5X,15HDO YOU WISH TO:,/,
     1       2X,41H1).  CONTINUE WITH THE PRESENT CONDITIONS,/,
     2       2X,23H2).  CHANGE MACH NUMBER,/,
     3       2X,20H3).  CHANGE ALTITUDE,/,
```

```
     4       2X,35H4).  CHANGE PERCENT OF FUEL LOADING,/)
      READ(6,*)OPTION

      IF(OPTION.LE.0 .OR. OPTION.GE.5)GOTO 10
      IF(OPTION.EQ.2)THEN
   31   WRITE(6,*)'  ENTER DESIRED MACH NUMBER'
        READ(6,*,ERR=31)MACH
        IF(MACH.LT.0.4 .OR. MACH.GT.0.8)THEN
          WRITE(6,*)'  MACH NUMBER CHOSEN IS OUT OF AERODYNAMIC ',
     1            'DATA RANGE: CHOOSE AGAIN'
          WRITE(6,*)' '
          GOTO 31
        ENDIF
        GOTO 10
      ELSEIF(OPTION.EQ.3)THEN
   32   WRITE(6,*)'  ENTER DESIRED ALTITUDE IN FEET '
        READ(6,*,ERR=32)ALT
        IF(ALT.LT.0.0 .OR. ALT.GT.60000.)THEN
          WRITE(6,*)'  ALTITUDE CHOSEN IS OUT OF RANGE: CHOOSE AGAIN'
          GOTO 32
        ENDIF
        GOTO 10
      ELSEIF(OPTION.EQ.4)THEN
   33   WRITE(6,*)'  ENTER DESIRED FUEL LOADING ( 0 TO 100 )'
        READ(6,*,ERR=33)FUEL
        IF(FUEL.LT.0.0 .OR. FUEL.GT.100.)THEN
          WRITE(6,*)'  FUEL LOADING DESIRED IS OUT OF RANGE'
          GOTO 33
        ENDIF
        GOTO 10
      ENDIF

 1100 FORMAT(///////,5X,39H******* WELCOME TO THE MANNED AIRCRAFT ,
     1       26HSIMULATION (MACS). *******,///,
     2       14X,47HTHE FOLLOWING SETUP MENU IS USED TO SET INITIAL,/,
     3       14X,43HAIRCRAFT CONFIGURATIONS, FLIGHT CONDITIONS,,/,
     4       14X,41HDAMAGE OPTIONS, AND HUMAN PILOT MODELLING,/,
     4       14X,8HOPTIONS.,////,
     6       14X,42HIT IS IMPORTANT THAT YOU SELECT ONLY THOSE,/,
     7       14X,43HOPTIONS PRESENTED TO REMAIN CONSISTENT WITH,/,
     8       14X,27HAVAILABLE AERODYNAMIC DATA.,////////)
 1110 FORMAT(14X,49HENTER DESIRED TOTAL FLIGHT SIMULATION TIME (sec).,/)
 1200 FORMAT(///,10X,22HPRESENT DAMAGE OPTIONS,//,5X,7HDAMAGE:,1X,A37)
 1205 FORMAT(9X,A37)
 1210 FORMAT(5X,21HTIME OF DAMAGE (sec):,2X,F5.2)
 1220 FORMAT(/,5X,15HDO YOU WISH TO:,
     1       /,2X,32H1).  CONTINUE WITH THESE OPTIONS,/
     2       2X,17H2).  MAKE CHANGES,/)
 1230 FORMAT(//5X,46HWHICH DAMAGE CONFIGURATION DO YOU WANT TO FLY?,/
     1       2X,33H0).  BASELINE AIRCRAFT, NO DAMAGE,//,
     2       2X,30H1).  33% VERTICAL TAIL REMOVED,//,
     3       2X,33H2).  50% RIGHT STABILATOR REMOVED,//,
     4       2X,33H3).  30% RIGHT WING PANEL REMOVED,//,
     5       2X,35H4).  30% STARBOARD WING DAMAGE PLUS,/,
     6       2X,37H          STARBOARD STABILATOR REMOVED,/)
 1240 FORMAT(//5X42HAT WHAT TIME TO YOU WANT DAMAGE TO OCCUR?,/,
     1       15X,21H(0.0 < time < ENDTME) ,/)

C       ************************************************************
```

```
C         *    SET TRAJECTORY COMMANDS BY CALLING SUBROUTINE SETTRJ   *
C         ************************************************************
!       CALL SETTRJ
!      G       ( .0001,
!      Y         NUMCMD , CMDS )
!       GAMMA = CMDS(1,3)
!       PHI = CMDS(1,2)
!       LOAD = 1.0 / COS( CMDS(1,2)*PI/180. )

  200 RETURN
      END
```

```
      SUBROUTINE PLTMOD
     G         ( MACH , ACMASS , IAC , QBAR ,
     G           U0 , ALPHA0 , BETA0 , DAMAGE , DATA ,
     Y           FOPT , GOPT , KLIMIT , ERRFLG )
C     ***************************************************************
C     *                                                             *
C     *                                                SEPT-24-91   *
C     *      SUBROUTINE PLTMOD                                      *
C     *     G         ( MACH , ACMASS , IAC , QBAR ,               *
C     *     G           U0 , ALPHA0 , BETA0 , DAMAGE , DATA ,      *
C     *     Y           FOPT , GOPT , KLIMIT , ERRFLG )            *
C     *                                                            *
C     *    DESCRIPTION :  OPTIMAL PILOT MODEL                      *
C     *                   ------------------                       *
C     *    THIS SUBROUTINE SETS UP THE OPTIMAL PILOT MODEL.        *
C     *    IT FIRST ASKS TO DEFINE THE MANOUVER TO BE FLOWN. THE   *
C     *    STEADY STATE VARIABLES ARE THEN CALCULATED AND THE      *
C     *    STABILITY DERIVATIVES DETERMINED.                       *
C     *    NEXT THE STATE MATRICES A, B, C AND F ARE CALCULATED    *
C     *    AND THE OPTIMUM CONTROLLER PROBLEM SOLVED (RICCATI-     *
C     *    EQUATION). FINALLY THE PERFORMANCE INDEX J(U) AS        *
C     *    WELL AS THE EXPECTED VALUES OF THE STATE VECTOR X AND   *
C     *    THE INPUT VECTOR U ARE CALCULATED.                      *
C     *                                                            *
C     *                                                            *
C     *    RESTRICTIONS                                            *
C     *         (NONE)                                             *
C     *                                                            *
C     *    COMMUNICATION                                           *
C     *         ARGUMENT LIST                                      *
C     *            GIVEN                                           *
C     *              MACH   (R) = MACH NUMBER (-)                  *
C     *              QBAR   (R) = DYNAMIC PRESSURE (lbs/ft^2)      *
C     *              IAC    (R) = INSTANTANEOUS AIRCRAFT MOMENT    *
C     *                           OF INERTIA TENSOR (lbs/ft^2)     *
C     *              ACMASS (R) = TOTAL AIRCRAFT MASS (slugs)      *
C     *              U0     (R) = AIRCRAFT VELOCITY (FT/S)         *
C     *              BETA0  (R) = SIDESLIP ANGLE                   *
C     *              DAMAGE (I) = NOT USED                         *
C     *              DATA   (I) = NOT USED                         *
C     *                                                            *
C     *            YIELDED                                         *
C     *              FOPT   (R) = NOT USED                         *
C     *              GOPT   (R) = NOT USED                         *
C     *              KLIMIT (R) = NOT USED                         *
C     *              ERRFLG (L) = ERROR FLAG                       *
C     *                                                            *
C     *                                                            *
C     *    ASSOCIATED SUBPROGRAMS                                  *
C     *         CALLED BY                                          *
C     *              PILMOD                                        *
C     *         CALLS TO                                           *
C     *              PLTDAT                                        *
C     *              SSMAN                                         *
C     *              PMADER                                        *
C     *              PIMAM                                         *
C     *              TRNPOS                                        *
C     *              MATMUL                                        *
C     *              RICSOL                                        *
C     *              GOFTAU                                        *
```

```
C        *              TRACE                                            *
C        *              INTEGRAL                                         *
C        *              MATSUB                                           *
C        *              LYAP                                             *
C        *                                                               *
C        *    REFERENCES                                                 *
C        *         (NONE)                                                *
C        *                                                               *
C        *    LANGUAGE                                                   *
C        *         STANDARD FORTRAN 77                                   *
C        *****************************************************************

      INTEGER NN,NR,NM,NL
      PARAMETER (NN=16,NR=3,NM=3,NE=3,NL=8,NA=10,NB=3)

C        ******************** ARGUMENT VARIABLES *********************
      INTEGER  KLIMIT , DAMAGE , DATA , IRIC , IGAIN
      REAL  MACH , ACMASS , IAC(3,3) , QBAR , U0 ,
     1      ALPHA0 , BETA0 , FOPT(NR,NN) , GOPT(NR,NM)
      LOGICAL  ERRFLG

C        ********************* COMMON VARIABLES **********************
C        *************************************************************
C        *           THE COMMON BLOCK /DRYDEN/ CONTAINS THE          *
C        *     THE CONSTANTS USED BY THE DRYDEN GUST VELOCITY MODEL   *
C        *************************************************************
      REAL  L0 , W(NE,NE),SIGMA0
      COMMON /DRYDEN/  L0, W, SIGMA0

C        *************************************************************
C        *           THE COMMON BLOCK /PLTMAT/ CONTAINS THE          *
C        *           MATRICIES USED TO MODEL THE HUMAN PILOT         *
C        *************************************************************
      REAL  A(NN,NN)  , B(NN,NR)  , C(NN,NN)  , E(NN,NE)  , H(NL,NN)  ,
     &      ATRAN(NN,NN) , BTRAN(NR,NN) , CTRAN(NN,NM) , HTRAN(NN,NL) ,
     &      PHIDT(NN,NN) , PHITDT(NN,NN), PHITAU(NN,NN), PHINTA(NN,NN),
     &      HOFDT(NN,NR)
      COMMON /PLTMAT/  A    , B    , C    , E    , H    ,
     &                 ATRAN  , BTRAN  , CTRAN  , HTRAN  ,
     &                 PHIDT  , PHITDT , PHITAU , PHINTA ,
     &                 HOFDT

C        *************************************************************
C        *     THE COMMON BLOCK /AEROP/ CONTAINS THE GENERALIZED     *
C        * AERODYNAMIC COEFFICIENT MATRICES WHOSE FORMS ARE VALID    *
C        * FOR BOTH CONVENTIONAL AND DAMAGED AIRCRAFT CONFIGURATIONS *
C        *    NOTE:                                                  *
C        *      IN ORDER TO ACCOMMODATED AIRCRAFT HAVING ARBITRARY   *
C        *    AERODYNAMICS, THE USE OF THE GENERALIZED AERODYNAMIC   *
C        *    COEFFRICIENT MATRICES REQUIRES THAT ALL COEFFICIENTS   *
C        *    HAVE IDENTICAL REFERENCE LENGTHS AND AREAS.            *
C        *************************************************************
      REAL  CFV(3,3), CFW(3,3), CFD(3,3), CMV(3,3), CMW(3,3), CMD(3,3),
     &      SREF , LREF , TAUE , TAUA , TAUR , KDELE , KDELA , KDELR
      COMMON /AEROP/
     &      CFV , CFW , CFD , CMV , CMW , CMD , SREF , LREF , TAUE ,
     &      TAUA , TAUR , KDELE , KDELA , KDELR
```

```
C       ********************** LOCAL VARIABLES **********************
      REAL  QDIAG(NM) , RDIAG(NR) , Q(NN,NN) , R(NR,NR) , W0(3,3) ,
     &      IACINV(3,3) , UTIL(3,3) , K(NN,NN),DT , TAU , LOAD
     &      C1 , C2 , C3 , TAUN , RINV(NR,NR),TNR(NN,NR),TNM(NN,NM),
     &      TMNN(NN,NN),TM(NN,NN),TMNE(NN,NE),TMNE1(NN,NE),BRB(NN,NN),
     &      TMEN(NE,NN), Z(NN,NN),QSTAR(NN,NN),
     &      EZ(NN,NN), EZTRAN(NN,NN),
     &      ETILD(NN,NE),ETRAN(NE,NN),QTILD(NN,NN),
     &      EWE(NN,NN),KEWE(NN,NN),CEWEC(NN,NN)
      REAL EEE(NN,NN),EXX(NN,NN),EUU(NR,NR),LOPT(NR,NN),TMRN(NR,NN),
     &      TMNR(NN,NR)
      REAL JUOPT,JU1,JU2
      INTEGER I,J,ANSWER,NITER,IFIRST
      CHARACTER*80 DUMMY
      CHARACTER*5 MODE
      character*6 cpara
      DATA IFIRST /0/

C
C       ******************* START EXECUTABLE CODE *******************
C


C       ****************************************************************
C       *                  MULTIVARIABLE PILOT MODEL                   *
C       *                                                              *
C       *            LINEAR QUADRATIC GAUSSIAN COMPENSATOR             *
C       *                                                              *
C       *   STATE = X(t) = (U/U0,BETA,ALPHA,P,Q,R,DELE,DELA,DELR,      *
C       *                   PHI,THETA,UG,VG,V1,WG,W1)                  *
C       *                                                              *
C       *   CONTROL = U(t) = (EL,AL,RU)                                *
C       *                                                              *
C       *   OUTPUT = Y(t) = (BETA,PHI,THETA)                           *
C       ****************************************************************


C       ****************************************************************
C       *      CALL PLTDAT TO READ IN PILOT MODEL AERODYNAMIC          *
C       *    COEFFICIENTS, REFERENCE AREAS AND LENGTH, ACTUATOR        *
C       *             CONSTANTS, AND GEARING CONSTANTS                 *
C       ****************************************************************
      CALL PLTDAT
     G          ( MACH , DAMAGE , DATA ,
     Y            KLIMIT , QDIAG , RDIAG , ERRFLG )
      IF(ERRFLG) GOTO 4000

C       ****************************************************************
C       *    INITIALIZE ARRAYS USED IN CALCULATING PILOT GAINS         *
C       ****************************************************************
      DO 30 I=1,NN
        DO 30 J=1,NN
   30     A(I,J) = 0.0
      DO 40 I=1,NN
        DO 40 J=1,NR
   40     B(I,J) = 0.0
      DO 45 I=1,NN
        DO 45 J=1,NE
   45     E(I,J) = 0.0
```

```
      DO 50 I=1,NN
        DO 50 J = 1,NN
   50     C(I,J) = 0.0
      DO 60 I=1,NL
        DO 60 J=1,NN
   60     H(I,J) = 0.0
      DO 80 I=1,NN
        DO 80 J=1,NN
   80    Q(I,J)=0.0
      DO 90 I=1,NR
        DO 90 J=1,NR
   90     R(I,J)=0.0
      DO 110 I=1,NR
  110   R(I,I)=RDIAG(I)
      R(1,1)=9.76e-3
      R(2,2)=4.28E-2
      r(3,3)=.1411
C        ***************************************************************
C        *  DEFINE STEADY STATE MANEUVER TO BE FLOWN AND CALCULATE     *
C        *  THE SS VARIABLES                                           *
C        ***************************************************************
      CALL SSMAN(QBAR,ACMASS,U0)

C        ***************************************************************
C        * CALCULATE THE AERODYNAMIQUE STABILITY DERIVATIVES FOR THE   *
C        * DEFINED MANEUVER. THE DERIVATIVES ARE INTERPOLATED FROM     *
C        * FLIGHT TEST DATA TABLE OR ESTIMATED (CF ROSKAM) WHEN THEY   *
C        * ARE NOT AVAILABLE                                           *
C        ***************************************************************
      CALL PMADER( MACH, ACMASS, IAC, QBAR, U0 )

C        ***************************************************************
C        * SET UP THE A-MATRIX                                         *
C        ***************************************************************
      CALL PIMAM(U0,IAC,A)

      A(7,7)=-1.0/TAUE
      A(8,8)=-1.0/TAUA
      A(9,9)=-1.0/TAUR

      10=1750.
      SIGMA0=6.
      A(12,12)=-U0/L0
      A(13,14)=(U0/(2*L0))**2
      A(14,14)=-U0/L0
      A(14,13)=-1
      A(15,16)=(U0/(2*L0))**2
      A(16,15)=-1
      A(16,16)=-U0/L0

! PROV OUTPUT OF A-MATRIX
!      DO I=1,NN
!        WRITE(*,'(10F8.4)')(A(I,J) ,J=1,NN)
!      END DO

C        ***************************************************************
C        *   CONSTRUCT THE CONTROL MATRIX B(I,J)                       *
C        ***************************************************************
      B(7,1) = KDELE/TAUE
      B(8,2) = KDELA/TAUA
```

```
      B(9,3) = KDELR/TAUR

C        ***************************************************************
C        *   CONSTRUCT THE NOISE MATRIX E(I,J) TO MULTIPLY  THE        *
C        *   THE GAUSSIAN WHITE NOISE.                                 *
C        *   DRYDEN TURBULENCE MODEL IN THE TIME DOMAINE FORM          *
C        ***************************************************************
      PI=4*ATAN(1.)
      SPSQR=SIGMA0*SQRT(2*U0/(PI*L0))
      E(12,1) =SPSQR
      E(13,2) =SQRT(3.)/2*SPSQR
      E(14,2) =1-SQRT(3.)*L0/U0*SPSQR
      E(15,3) =SQRT(3.)/2*SPSQR
      E(16,3) =1-SQRT(3.)*L0/U0*SPSQR

C        ***************************************************************
C        *       CONSTRUCT THE AIRCRAFT OUTPUT MATRIX  C(I,J)          *
C        *              Y(t)=( BETA, PHI, GAMMA)                       *
C        ***************************************************************
      DO I=1,NN
        C(I,I)=1.
      ENDDO
C        ***************************************************************
C        *   CONSTRUCT THE OUTPUT WEIGHTING MATRIX Q(I,J)              *
C        ***************************************************************
      Q(2,2)=205
      Q(10,10)=131
      Q(11,11)=365
C        ***************************************************************
C        *        TRANSPOSE A,B,C MATRICIES FOR FUTURE USE             *
C        ***************************************************************
      CALL TRNPOS(A,NN,NN,ATRAN)
      CALL TRNPOS(B,NN,NR,BTRAN)
      CALL TRNPOS(C,NN,NN,CTRAN)

C     ***************************************************************
C     *    SOLUTION OF THE ALGEBRAIC RICCATI EQUATION                *
C     *                                                              *
C     *                        T       -1  T      T                 *
C     *            0 = K*A+A K-K*B*R  *B *K - C *Q*C                 *
C     ***************************************************************

      CALL MATMUL(NN,NN,CTRAN,NN,NN,Q,NN,NN,TMNN,IER)
        IF (IER) GOTO 6000
      CALL MATMUL(NN,NN,TMNN,NN,NN,C,NN,NN,Q,IER)

      CALL RICSOL(A,B,Q,R,K,errflg)
          IF (ERRFLG) GO TO 6000

C        ***************************************************************
C        *   CALCULATION OF THE VALUE OF THE PERFORMANCE INDEX FOR     *
C        *   THE SYSTEM WITH A TIME LAG TAU AND WHITE NOISE WITH       *
C        *   COVARIANCE MATRIX W AS INPUT.                             *
C        *                                                             *
C        *                                Z*s        T  T   Z*s T      *
C        *   J(U) = tr(Q * integral(e   *C*E*W*E *C *(e    ) ds)      *
C        *                                                             *
C        *                          T                                  *
C        *          + tr(K*E*W*E )                                     *
C        *                                                             *
```

```
C        *                          -1                                  *
C        *   WHERE:      Z = C*A*C                                      *
C        *                    -1  Z*tau                                 *
C        *                 E = C  *e     *C*E                           *
C        *                                                              *
C        ****************************************************************

      N11=11
C        ****************************************************************
C        *                                    A*TAU                     *
C        *  CALCULATE TRANSITIONMATRIX   e                              *
C        ****************************************************************
      TAU = .1           ! prov
      CALL GOFTAU(A,TAU,EZ,I,ERRFLG)
      CALL TRNPOS(EZ,NN,NN,EZTRAN)

C        ****************************************************************
C        *                    T                                         *
C        *  CALCULATE E*W*E                                             *
C        ****************************************************************
      W(1,1)=1.
      W(2,2)=1.
      W(3,3)=1.
      CALL TRNPOS(E,NN,NE,ETRAN)
      CALL MATMUL(NN,NE,E,NE,NE,W,NN,NE,TMNE,ERRFLG)
          IF (ERRFLG) GOTO 5000
      CALL MATMUL(NN,NE,TMNE,NE,NN,ETRAN,NN,NN,EWE,ERRFLG)
          IF (ERRFLG) GOTO 5000

C        ****************************************************************
C        *  CALCULATE SECOND TERM OF THE PERFORMANCE INTGRAL WTD        *
C        ****************************************************************
      CALL MATMUL(NN,NN,EZ,NN,NN,EWE,NN,NN,TMNN,ERRFLG)
      CALL MATMUL(NN,NN,TMNN,NN,NN,EZTRAN,NN,NN,TM,ERRFLG)
      CALL MATMUL(NN,NN,K,NN,NN,TM,NN,NN,KEWE,ERRFLG)
          IF (ERRFLG) GOTO 5000
      CALL TRACE(KEWE,NN,JU2)

C        ****************************************************************
C        *  CALCULATE FIRST TERM OF THE PERFORMANCE INTEGRAL WTD        *
C        ****************************************************************
      CALL INTEGRAL(A,NN,N11,EWE,TAU,eee)
      CALL MATMUL(NN,NN,Q,NN,NN,EEE,NN,NN,TM,ERRFLG)
      CALL TRACE(TM,NN,JU1)

C        ****************************************************************
C        *  THE PERFORMANCE INTGRAL WITH TIME DELAY                     *
C        ****************************************************************
      JUOPT=JU1+JU2
      PRINT*,JU1,JU2,JUOPT

C        ****************************************************************
C        *  THE PERFORMANCE INTEGRAL WITHOUT TIME DELAY                 *
C        ****************************************************************
      CALL MATMUL(NN,NN,K,NN,NN,EWE,NN,NN,KEWE,ERRFLG)
      CALL TRACE(KEWE,NN,JUOPT)
      PRINT*,JUOPT

!C       ****************************************************************
!C       *  CALCULATION OF MEAN SQUARE VALUE OF X AND U        WITH TIME    *
```

```
!C        * DELAY                                                        *
!C        ****************************************************************
!
      DO I=1,3
         RINV(I,I)=1./R(I,I)
      ENDDO
      CALL MATMUL(NR,NN,BTRAN,NN,NN,K,NR,NN,TMRN)
      CALL MATMUL(NR,NR,RINV,NR,NN,TMRN,NR,NN,LOPT)
!
!C                T
!C       *** E{ee }  EEE ***
!
!       DO I=1,NN
!         WRITE(6,'(2f8.3)')EXX(I,I),EEE(I,I)
!       ENDDO
!C              T
!C       *** E{uu }  EUU ***
!       CALL MATMUL(NR,NN,LOPT,NN,NN,EXX,NR,NN,TMRN)
!       CALL TRNPOS(LOPT,NR,NN,TMNR)
!       CALL MATMUL(NR,NN,TMRN,NN,NR,TMNR,NR,NR,EUU)
!       DO I=1,NR
!          WRITE(6,'(F10.4)')EUU(I,I)
!       ENDDO

C         ***************************************************************
C         *  CALCULATION OF MEAN SQUARE VALUE OF X AND U WITHOUT TIME *
C         * DELAY                                                        *
C         ***************************************************************

C                T
C       *** E{xx }  EXX ***

      CALL MATMUL(NN,NR,B,NR,NN,LOPT,NN,NN,TMNN)
      CALL MATSUB(NN,NN,A,tmnn,tm)
      CALL LYAP(NN,NN,nN,tm,EWE,exx)
      DO I=1,NN
c        WRITE(6,'(2f15.3)')EXX(I,I)
         print*,'exx',I,exx(i,i)
      ENDDO
      print*,' '
C              T
C       *** E{uu }  EUU ***
      CALL MATMUL(NR,NN,LOPT,NN,NN,EXX,NR,NN,TMRN)
      CALL TRNPOS(LOPT,NR,NN,TMNR)
      CALL MATMUL(NR,NN,TMRN,NN,NR,TMNR,NR,NR,EUU)
      DO I=1,NR
c         WRITE(6,'(F10.4)')EUU(I,I)
          print*,'euu',i,euu(i,i)
      ENDDO
C        ***************************************************************
C        *                      OPEN OUTPUT FILE                       *
C        ***************************************************************
      IF (CFLAG.NE.0)GOTO 15
      CFLAG=1
 10   OPEN(UNIT=13,FILE='PERF.DAT',STATUS='NEW')
      OPEN(UNIT=14,FILE='PERF.CSV',STATUS='NEW')
      WRITE(13,'(/,1X,''STATE WEIGHTING MATRIX Q'',/)')
      DO I=1,NN
        WRITE(13,'(16F6.1)')(Q(I,J),J=1,NN)
      ENDDO
```

```
      WRITE(13,'(/,1X,''CONTROL WEIGHTING MATRIX R'',/)')
      DO I= 1,3
        WRITE(13,'(3E13.5)')(R(I,J),J=1,3)
      ENDDO
      print*,'enter parameter description (max 6 char.)'
      read*,cpara
      WRITE(13,'(//,1x,a6,6X,''beta'',9X,''phi'',8X,''theta'',
     1 10X,''ELEV'',9X,''AIL'',9X,''RUD'',11X,''J(u)'',/)')cpara

 15   continue
      print*,'Enter parameter'
      read*,rpara
      WRITE(13,111)
     1  rpara,EXX(2,2),EXX(10,10),EXX(11,11),
     2  (EUU(I,I),I=1,3),JUOPT
      WRITE(14,111)
     1  rpara,EXX(2,2),EXX(10,10),EXX(11,11),
     2  (EUU(I,I),I=1,3),JUOPT
 111   FORMAT(F6.2,',',E12.4,',',E12.4,',',E12.4,',',
     1 E12.4,',',E12.4,',',E12.4,',',E12.4)
C      CLOSE(UNIT=13,STATUS='KEEP')

      RETURN

 4000 write(6,*)'  PLTMOD:  ERROR IN PLTDAT'
      RETURN

 5000 write(6,*)'  PLTMOD:  ERROR IN MATMUL'
      RETURN

 6000 write(6,*)'  PLTMOD:  ERROR IN RICCATI'
      RETURN

      END
```

```
      SUBROUTINE SSMAN(
     G                  QBAR, ACMASS, U0)
C     ****************************************************************
C     *                                                              *
C     *                                                 SEPT-24-91   *
C     *    CALLING SEQUENCE                                          *
C     *        SUBROUTINE SSMAN                                      *
C     *   G          ( QBAR, ACMASS, U0)                             *
C     *                                                              *
C     *    DESCRIPTION :  STEADY STATE MANEUVER                      *
C     *                   --------------------                       *
C     *    THIS SUBROUTINE READS IN WHAT IS THE DESIRED MANEUVEI     *
C     *    THE OPTIONS IMPLEMENTED SO FAR ARE:                       *
C     *              - HORIZONTAL UNACCELAREATED FLIGHT              *
C     *              - HORIZONTAL STEADY TURN                        *
C     *              - SYMMETRICAL PULLUP                            *
C     *    WHERE NECESSARY THE VARIABLES DEFINING THE MANEUVER       *
C     *    ARE THEN READ IN.                                         *
C     *                                                              *
C     *    RESTRICTIONS                                              *
C     *         (NONE)                                               *
C     *                                                              *
C     *    COMMUNICATION                                             *
C     *         ARGUMENT LIST                                        *
C     *            GIVEN                                             *
C     *              QBAR   (R) = DYNAMIC PRESSURE (lbs/ft^2)        *
C     *              ACMASS (R) = TOTAL AIRCRAFT MASS (slugs)        *
C     *              U0     (R) = AIRCRAFT VELOCITY (FT/S)           *
C     *            YIELDED                                           *
C     *              (NONE)                                          *
C     *                                                              *
C     *    LOCAL VARIABLES                                           *
C     *              ANS    (I) = SELECTION VARIABLE                 *
C     *              G      (R) = GRAVITY CONSTANT                   *
C     *              GTOR   (R) = CONVERSION FACTOR RAD TO DEG       *
C     *              LOAD   (R) = LOAD FACTOR                        *
C     *                                                              *
C     *    COMMON VARIABLES                                          *
C     *              P0     (R) = AIRCRAFT ROLL RATE WRITTEN IN      *
C     *                           THE AIRCRAFT FRAME (rad/sec)       *
C     *              Q0     (R) = AIRCRAFT PITCH RATE WRITTEN IN     *
C     *                           THE AIRCRAFT FRAME (rad/sec)       *
C     *              R0     (R) = AIRCRAFT YAW RATE WRITTEN IN       *
C     *                           THE AIRCRAFT FRAME (rad/sec)       *
C     *              PHI0   (R) = SS BANK ANGLE                      *
C     *              PSI0   (R) = SS YAW ANGLE                       *
C     *              THETA0 (R) = SS PITCH ANGLE                     *
C     *              LIFT   (R) = LIFT NEEDED FOR GIVEN FLIGHT       *
C     *                           CONDITION (lbs)                    *
C     *                                                              *
C     *    ERROR CHECKS                                              *
C     *         (NONE)                                               *
C     *                                                              *
C     *    ASSOCIATED SUBPROGRAMS                                    *
C     *         CALLED BY                                            *
C     *              PLTMOD                                          *
C     *         CALLS TO                                             *
C     *              (NONE)                                          *
C     *                                                              *
C     *    REFERENCES                                                *
```

```
C         *          (NONE)                                              *
C         *                                                              *
C         *     LANGUAGE                                                 *
C         *          STANDARD FORTRAN 77                                 *
C         ************************************************************

C         ************************************************************
C         *     THE COMMON BLOCK  /SSVAR/  CONTAINS THE STEADY STATE   *
C         * STATE VARIABLES CALCULATED FOR THE GIVEN MANEUVER.         *
C         ************************************************************
      REAL    P0,Q0,R0
      REAL    PHI0,PSI0,THETA0
      REAL    LIFT
      COMMON /SSVAR/
     1        P0,Q0,R0,
     2        PHI0,PSI0,THETA0,
     3        LIFT

C
C         ******************** ARGUMENT VARIABLES *********************
C
      REAL    QBAR,ACMASS,U0

C
C         ******************* LOCAL VARIABLES ************************
C
      REAL LOAD,G,DTOR
      INTEGER ANS
C         ********************* FORMAT STATEMENTS *********************

 1000 FORMAT(1X,'SELECT DESIRED MANEUVER:',/,
     1          1X,'THE OPTIONS ARE...'
     2         ,//,T5,'1. HORIZONTAL UNACCELERATED FLIGHT',
     3          /,T5,'2. HORIZINTAL STEADY TURN',
     4          /,T5,'3. STEADY PULLUP',//)
 1010 FORMAT(I1)
 1020 FORMAT(1X,'THIS IS NOT AN ACCEPTABLE CHOICE!')
 1030 FORMAT(1X,'YOUR INPUT OPTIONS ARE ...',/,
     1                            25X,'RATE OF TURN   1',/,
     2                            25X,'BANK ANGLE     2',/,
     3                            25X,'LOAD FACTOR    3',/,
     4                              1X,'ENTER OPTION...')
 1040 FORMAT(1X,'ENTER TURN RATE...   [DEG/S]')
 1050 FORMAT(1X,'ENTER BANK ANGLE...  [DEG]')
 1060 FORMAT(1X,'ENTER LOAD FACTOR... [-]')
 1070 FORMAT(1X,'ENTER THETA0... [DEG]')

C
C         ******************* START EXECUTABLE CODE *******************
C
      G=32.17
      DTOR=1./57.3
 20   WRITE(6,1000)
      READ(5,1010,ERR=20)ANS
      IF(ANS.EQ.1)GOTO 200
      IF(ANS.EQ.2)GOTO 300
      IF(ANS.EQ.3)GOTO 400
      WRITE(6,1020)
      GOTO 20
```

```
C
C  HORIZONTAL UNACCELERATED FLIGHT
C
 200  P0 = 0.
      Q0 = 0.
      R0 = 0.
      PHI0 = 0.
      THETA0 = 0.
      PSI0 = 0.
      LOAD = 1.
      GOTO 900
C
C  HORIZONTAL STEADY TURN
C
 300  WRITE(6,1030)
      READ(5,1010,ERR=300)ANS
      IF(ANS.EQ.1)GOTO 320
      IF(ANS.EQ.2)GOTO 330
      IF(ANS.EQ.3)GOTO 340
      WRITE(6,1020)
      GOTO 300

 320  WRITE(6,1040)
      READ(5,*,ERR=320)PSIDOT
      PSIDOT=PSIDOT*DTOR
      PHI0=ATAN(PSIDOT*U0/G)
      R0=PSIDOT*COS(PHI0)
      Q0=PSIDOT*SIN(PHI0)
      LOAD=1./COS(PHI0)
      GOTO 350

 330  WRITE(6,1050)
      READ(5,*,ERR=330)PHI0
      PHI0=PHI0*DTOR
      R0=G*SIN(PHI0)/U0
      Q0=R0*TAN(PHI0)
      LOAD=1./COS(PHI0)
      GOTO 350

 340  WRITE(6,1060)
      READ(5,*,ERR=340)LOAD
      Q0=G/U0*(LOAD-1./LOAD)
      R0=G/U0*SQRT(1.-1./LOAD**2)
      PHI0=ACOS(1./LOAD)
      GOTO 350

 350  P0=0.
      THETA0=0.
      GOTO 900
C
C   SYMMETRICAL PULLUP
C
 400  WRITE(6,1060)
      READ(5,*,ERR=400)LOAD
 410  WRITE(6,1070)
      READ(5,*,ERR=400)THETA0
      THETA0=THETA0*DTOR
      P0=0.
      R0=0.
      PHI0=0.
```

```
      PSI0=0.
      Q0=G/U0*(COS(THETA0)-LOAD)

900   LIFT = ACMASS*G*LOAD
      print*,lift
      RETURN
      END
```

```
      SUBROUTINE PMADER
     G     ( MACH, ACMASS, IAC, QBAR, U0)
C     ***************************************************************
C     *                                                 SEPT 11, 1991 *
C     *                                                               *
C     *    CALLING SEQUENCE                                           *
C     *        SUBROUTINE PMADER                                      *
C     *             ( MACH, ACMASS, IAC, QBAR, U0 )                   *
C     *                                                               *
C     *    DESCRIPTION : AERODYNAMIC DERIVATIVES: FLIGHT TEST DATA    *
C     *                  ----------- -----------  ------ ---- ----    *
C     *        SUBROUTINE MADER DETERMINES THE AERODYNAMIC DERIV-      *
C     *    ATIVES FOR THE SPECIFIED STEADY STATE FLIGHT MANEUVER.     *
C     *    THE FLIGHT TEST DATA TABLE IS USED.                        *
C     *                                                               *
C     *    RESTRICTIONS                                               *
C     *        (NONE)                                                 *
C     *                                                               *
C     *    COMMUNICATION                                              *
C     *        ARGUMENT LIST                                          *
C     *            GIVEN                                              *
C     *           MACH    (R) = CURRENT AIRCRAFT MACH NUMBER          *
C     *           ACMASS  (R) = AIRCRAFT MASS (SLUGS)                 *
C     *           IAC     (R) = THE INSTANTANEOUS AIRCRAFT MOMENT     *
C     *                         OF INERTIA TENSOR (lbs/ft^2)          *
C     *           U0      (R) = REFERENCE VELOCITY ABOUT WHICH        *
C     *                         THE SMALL PERTURBATION MODEL IS       *
C     *                         BASED.                                *
C     *              YIELDED                                          *
C     *           (NONE)                                              *
C     *                                                               *
C     *    LOCAL VARIABLES                                            *
C     *           PI      (R) = THE CONSTANT 3.14159....              *
C     *           P       (R) = AIRCRAFT ROLL RATE WRITTEN IN THE     *
C     *                         AIRCRAFT FRAME (rad/sec)              *
C     *           Q       (R) = AIRCRAFT PITCH RATE WRITTEN IN THE    *
C     *                         AIRCRAFT FRAME (rad/sec)              *
C     *           R       (R) = AIRCRAFT YAW RATE WRITTEN IN THE      *
C     *                         AIRCRAFT FRAME (rad/sec)              *
C     *           SPAN    (R) = AIRCRAFT WINGSPAN TO WHICH THE        *
C     *                         AERODYNAMIC DATA IS REFERENCED (ft)   *
C     *           CHORD   (R) = AIRCRAFT MEAN AERODYNAMIC CHORD       *
C     *                         LENGTH TO WHICH THE AERODYNAMIC       *
C     *                         DATA IS REFERENCED (ft)               *
C     *           AREA    (R) = AIRCRAFT WING AREA TO WHICH THE       *
C     *                         AERODYNAMIC DATA IS REFERENCED        *
C     *                         (ft^2)                                *
C     *           RALPHA  (R) = ANGLE OF ATTACK (rad)                 *
C     *           LOW     (R) = TEMPORARY STORAGE USED IN TWO         *
C     *                         DIMENSIONAL INTERPOLATIONS            *
C     *           HIGH    (R) = TEMPORARY STORAGE USED IN TWO         *
C     *                         DIMENSIONAL INTERPOLATIONS            *
C     *           NM      (I) = NUMBER OF DATA POINTS IN MACH         *
C     *                         NUMBER TABLE                          *
C     *           NQ      (I) = NUMBER OF DATA POINTS IN DYNAMIC      *
C     *                         PRESSURE TABLE                        *
C     *           NA      (I) = NUMBER OF DATA POINTS IN ANGLE OF     *
C     *                         ATTACK TABLE                          *
C     *           NE      (I) = NUMBER OF DATA POINTS IN ELEVATOR     *
C     *                         DEFLECTION TABLE                      *
```

```
C     *          NL      (I) = NUMBER OF DATA POINTS IN CL^2      *
C     *                        TABLE                              *
C     *          KM      (I) = LARGEST DATA POINT IN MACH NUMBER  *
C     *                        TABLE (M) WHICH IS LESS THAN OR    *
C     *                        EQUAL TO THE CURRENT MACH NUMBER   *
C     *                        VALUE (MACH)                       *
C     *          KM1     (I) = KM + 1                             *
C     *          MPRCNT  (R) = LOCATION OF CURRENT MACH NUMBER    *
C     *                        (MACH) BETWEEN TWO DATA POINTS IN  *
C     *                        THE MACH NUMBER TABLE (M)          *
C     *          KQ      (I) = LARGEST DATA POINT IN DYNAMIC      *
C     *                        PRESSURE TABLE (QD) WHICH IS LESS  *
C     *                        THAN OR EQUAL TO THE CURRENT       *
C     *                        DYNAMIC PRESSURE VALUE (QBAR)      *
C     *          KQ1     (I) = KQ + 1                             *
C     *          QPRCNT  (R) = LOCATION OF CURRENT DYNAMIC        *
C     *                        PRESSURE (QBAR) BETWEEN TWO DATA   *
C     *                        POINTS IN THE DYNAMIC PRESSURE     *
C     *                        TABLE (QD)                         *
C     *          KE      (I) = LARGEST DATA POINT IN ELEVATOR     *
C     *                        DEFLECTION TABLE (E) WHICH IS LESS *
C     *                        THAN OR EQUAL TO THE CURRENT       *
C     *                        ELEVATOR DEFLECTION VALUE (DELEV)  *
C     *          KE1     (I) = KE + 1                             *
C     *          EPRCNT  (R) = LOCATION OF CURRENT ELEVATOR       *
C     *                        DEFLECTION (DELEV) BETWEEN TWO DATA*
C     *                        POINTS IN THE ELEVATOR DEFLECTION  *
C     *                        TABLE (E)                          *
C     *          KA      (I) = LARGEST DATA POINT IN ANGLE OF     *
C     *                        ATTACK TABLE (A) WHICH IS LESS     *
C     *                        THAN OR EQUAL TO THE CURRENT ANGLE *
C     *                        OF ATTACK (ALPHA)                  *
C     *          KA1     (I) = KA + 1                             *
C     *          APRCNT  (R) = LOCATION OF CURRENT ANGLE OF ATTACK*
C     *                        (ALPHA) BETWEEN TWO DATA POINTS IN *
C     *                        THE ANGLE OF ATTACK TABLE (A)      *
C     *          KL      (I) = LARGEST DATA POINT IN COEFFICIENT  *
C     *                        OF LIFT SQUARED TABLE (CL2) WHICH  *
C     *                        IS LESS THAN OR EQUAL TO THE       *
C     *                        CURRENT LIFT COEFFICIENT SQUARED   *
C     *                        (CLIFT2)                           *
C     *          KL1     (I) = KL + 1                             *
C     *          LPRCNT  (R) = LOCATION OF CURRENT LIFT COEF.     *
C     *                        SQUARED (CLIFT2) BETWEEN TWO DATA  *
C     *                        POINTS IN THE LIFT COEFFICIENT     *
C     *                        SQUARED TABLE (CL2)                *
C     *          CLIFT   (R) = TOTAL LIFT COEFFICIENT WRITTEN IN  *
C     *                        STABILITY AXES                     *
C     *          CLIFT2  (R) = TOTAL LIFT COEFFICIENT SQUARED     *
C     *          CLFT0   (R) = ZERO ANGLE OF ATTACK LIFT COEF.    *
C     *          CLFTA   (R) = LIFT CURVE DERIVATIVE DUE TO ANGLE *
C     *                        OF ATTACK                          *
C     *          CLTRM   (R) = LIFT COEFFICIENT NEEDED FOR TRIM   *
C     *                        FLIGHT CONDITIONS                  *
C     *          CLFTE   (R) = LIFT CURVE CONTROL DERIVATIVE DUE  *
C     *                        TO ELEVATOR DEFLECTION             *
C     *          CD      (R) = TOTAL DRAG COEFFICIENT WRITTEN IN  *
C     *                        STABILITY AXES                     *
C     *          CDU     (R) = DRAG DERIVATIVE DUE TO VELOCITY    *
C     *          CDTRM   (R) = DRAG COEFFICIENT IN TRIM FLIGHT    *
```

```
C     *          CDSB    (R) = ADDITIONAL DRAG COEFFICIENT DUE TO *
C     *                        DEPLOYMENT OF SPEED BRAKES         *
C     *          CYB     (R) = SIDE FORCE DERIVATIVE DUE TO       *
C     *                        SIDESLIP ANGLE (BETA)              *
C     *          CYDR    (R) = SIDE FORCE DERIVATIVE DUE TO       *
C     *                        RUDDER DEFLECTION (DRUD)           *
C     *          CLB     (R) = ROLLING MOMENT DERIVATIVE DUE TO   *
C     *                        SIDESLIP ANGLE (BETA)              *
C     *          CLR     (R) = ROLLING MOMENT DERIVATIVE DUE TO   *
C     *                        YAW RATE (R)                       *
C     *          CLP     (R) = ROLLING MOMENT DERIVATIVE DUE TO   *
C     *                        ROLL RATE (P)                      *
C     *          CLDR    (R) = ROLLING MOMENT DERIVATIVE DUE TO   *
C     *                        RUDDER DEFLECTION (DRUD)           *
C     *          CLDA    (R) = ROLLING MOMENT DERIVATIVE DUE TO   *
C     *                        AILERON DEFLECTION (DAIL)          *
C     *          CM      (R) = TOTAL PITCHING MOMENT COEFFICIENT  *
C     *          CM0     (R) = PITCHING MOMENT COEFFICIENT        *
C     *                        WITH NO CONTROL DEFLECTIONS AND    *
C     *                        ZERO SIDESLIP AND ANGLE OF ATTACK  *
C     *          CMA     (R) = PITCHING MOMENT DERIVATIV DUE TO   *
C     *                        ANGLE OF ATTACK                    *
C     *          CMDE    (R) = PITCHING MOMENT DERIVATIVE DUE TO  *
C     *                        ELEVATOR DEFLECTION (DELEV)        *
C     *          CMQ     (R) = PITCHING MOMENT DERIVATIVE DUE TO  *
C     *                        PITCH RATE (Q)                     *
C     *          CNB     (R) = YAWING MOMENT DERIVATIVE DUE TO    *
C     *                        SIDESLIP ANGLE (BETA)              *
C     *          CNR     (R) = YAWING MOMENT DERIVATIVE DUE TO    *
C     *                        YAW RATE (R)                       *
C     *          CNP     (R) = YAWING MOMENT DERIVATIVE DUE TO    *
C     *                        ROLL RATE (P)                      *
C     *          CNDR    (R) = YAWING MOMENT DERIVATIVE DUE TO    *
C     *                        RUDDER DEFLECTION (DRUD)           *
C     *          CNDA    (R) = YAWING MOMENT DERIVATIVE DUE TO    *
C     *                        AILERON DEFLECTION (DAIL)          *
C     *          NMA     (R) = CORRECTION TERM TO CMA DUE TO NON- *
C     *                        LINEARITIES IN COEFFICIENT DATA    *
C     *                        ATTRIBUTED TO DYNAMIC PRESSURE     *
C     *          NME     (R) = CORRECTION TERM TO CMDE DUE TO NON-*
C     *                        LINEARITIES IN COEFFICIENT DATA    *
C     *                        ATTRIBUTED TO DYNAMIC PRESSURE     *
C     *          NDRC    (R) = CORRECTION TERM TO CYDR AND CNDR   *
C     *                        DUE TO NON-LINEARITIES IN          *
C     *                        COEFFICIENT DATA ATTRIBUTED TO     *
C     *                        DYNAMIC PRESSURE                   *
C     *          KDE     (R) = CORRECTION TERM TO CLFTE DUE TO    *
C     *                        NON-LINEARITIES IN COEFFICIENT DATA*
C     *                        ATTRIBUTED TO MACH NUMBER AND      *
C     *                        ELEVATOR DEFLECTION                *
C     *          DETRIM  (R) = ELEVATOR DEFLECTION NEEDED FOR     *
C     *                        TRIM FLIGHT                        *
C     *          KDETRM  (R) = CORRECTION TERM TO CLFTE AT TRIM   *
C     *                        FLIGHT CONDITIONS DUE TO           *
C     *                        NON-LINEARITIES IN COEFFICIENT DATA*
C     *                        ATTRIBUTED TO MACH NUMBER AND      *
C     *                        ELEVATOR DEFLECTION                *
C     *                                                           *
C     *     COMMON VARIABLES                                      *
C     *          OMEGA   (R) = AIRCRAFT ANGULAR VELOCITY VECTOR   *
```

```
C     *                    WRITTEN IN THE AIRCRAFT FRAME      *
C     *                    (rad/sec)                          *
C     *          DELEV  (R) = SYMMETRICAL DEFLECTION OF THE   *
C     *                    HORIZONTAL TAIL SURFACE (deg)      *
C     *          DRUD   (R) = RUDDER DEFLECTION (deg)         *
C     *          DAIL   (R) = ASYMMETRIC DEFLECTION OF THE    *
C     *                    HORIZONTAL TAIL SURFACE (deg)      *
C     *          DSPOIL (R) = SPOILER DEFLECTION (deg)        *
C     *          DSB    (R) = SPEED BRAKE DEPLOYMENT          *
C     *          M      (R) = TABLE OF MACH NUMBER VALUES     *
C     *          QD     (R) = TABLE OF DYNAMIC PRESSURE VALUES*
C     *          E      (R) = TABLE OF ELEVATOR DEFLECTION VALUES*
C     *          A      (R) = TABLE OF ANGLE OF ATTACK VALUES *
C     *          CL2    (R) = TABLE OF LIFT COEFFICIENT SQUARED *
C     *                    VALUES                             *
C     *          ACLFT0 (R) = INTERPOLATION TABLE FOR CLFT0 TERM *
C     *          ACLFTA (R) = INTERPOLATION TABLE FOR CLFTA TERM *
C     *          ACLFTE (R) = INTERPOLATION TABLE FOR CLFTE TERM *
C     *          ACDTRM (R) = INTERPOLATION TABLE FOR CDTRM TERM *
C     *          ACDSB  (R) = INTERPOLATION TABLE FOR CDSB TERM  *
C     *          ACYB   (R) = INTERPOLATION TABLE FOR CYB TERM   *
C     *          ACYDR  (R) = INTERPOLATION TABLE FOR CYDR TERM  *
C     *          ACLB   (R) = INTERPOLATION TABLE FOR CLB TERM   *
C     *          ACLR   (R) = INTERPOLATION TABLE FOR CLR TERM   *
C     *          ACLP   (R) = INTERPOLATION TABLE FOR CLP TERM   *
C     *          ACLDR  (R) = INTERPOLATION TABLE FOR CLDR TERM  *
C     *          ACLDA  (R) = INTERPOLATION TABLE FOR CLDA TERM  *
C     *          ACLDSP (R) = INTERPOLATION TABLE FOR CLDSP TERM *
C     *          ACM0   (R) = INTERPOLATION TABLE FOR CM0 TERM   *
C     *          ACMA   (R) = INTERPOLATION TABLE FOR CMA TERM   *
C     *          ACMDE  (R) = INTERPOLATION TABLE FOR CMDE TERM  *
C     *          ACMQ   (R) = INTERPOLATION TABLE FOR CMQ TERM   *
C     *          ACNB   (R) = INTERPOLATION TABLE FOR CNB TERM   *
C     *          ACNR   (R) = INTERPOLATION TABLE FOR CNR TERM   *
C     *          ACNP   (R) = INTERPOLATION TABLE FOR CNP TERM   *
C     *          ACNDR  (R) = INTERPOLATION TABLE FOR CNDR TERM  *
C     *          ACNDSP (R) = INTERPOLATION TABLE FOR CNDSP TERM *
C     *          ACNDA  (R) = INTERPOLATION TABLE FOR CNDA TERM  *
C     *          NMASUB (R) = SUBSONIC TABLE OF NMA TERM         *
C     *          NMASUP (R) = SUPERSONIC TABLE OF NMA TERM       *
C     *          NMESUB (R) = SUBSONIC TABLE OF NME TERM         *
C     *          NMESUP (R) = SUPERSONIC TABLE OF NME TERM       *
C     *          ANDR   (R) = INTERPOLATION TABLE FOR NDR TERM   *
C     *          AKDE   (R) = INTERPOLATION TABLE FOR KDE AND    *
C     *                    KDETRM TERMS                       *
C     *          XU     (R) = X-FORCE DERIVATIVE DUE TO VELOCITY *
C     *          XA     (R) = X-FORCE DERIVATIVE DUE TO ANGLE OF *
C     *                    ATTACK                             *
C     *          YB     (R) = SIDE FORCE DERIVATIVE DUE TO SIDE- *
C     *                    SLIP ANGLE                         *
C     *          YDR    (R) = SIDE FORCE DERIVATIVE DUE TO RUDDER*
C     *                    DEFLECTION                         *
C     *          ZU     (R) = Z-FORCE DERIVATIVE DUE TO VELOCITY *
C     *          ZA     (R) = Z-FORCE DERIVATIVE DUE TO ANGLE OF *
C     *                    ATTACK                             *
C     *          ZDE    (R) = Z-FORCE DERIVATIVE DUE TO ELEVATOR *
C     *                    DEFLECTION                         *
C     *          LB     (R) = ROLLING MOMENT DERIVATIVE DUE TO   *
C     *                    SIDESLIP ANGLE                     *
C     *          LP     (R) = ROLLING MOMENT DERIVATIVE DUE TO   *
```

```
C       *                          ROLL RATE                            *
C       *           LR     (R) = ROLLING MOMENT DERIVATIVE DUE TO       *
C       *                          YAW RATE                             *
C       *           LDR    (R) = ROLLING MOMENT DERIVATIVE DUE TO       *
C       *                          RUDDER DEFLECTION                    *
C       *           LDA    (R) = ROLLING MOMENT DERIVATIVE DUE TO       *
C       *                          AILERON DEFLECTION                   *
C       *           MU     (R) = PITCHING MOMENT DERIVATIVE DUE TO      *
C       *                          VELOCITY                             *
C       *           MA     (R) = PITCHING MOMENT DERIVATIVE DUE TO      *
C       *                          ANGLE OF ATTACK                      *
C       *           MADOT  (R) = PITCHING MOMENT DERIVATIVE DUE TO      *
C       *                          ANGLE OF ATTACK CHANGE RATE          *
C       *           MQ     (R) = PITCHING MOMENT DERIVATIVE DUE TO      *
C       *                          PITCH RATE                           *
C       *           MDE    (R) = PITCHING MOMENT DERIVATIVE DUE TO      *
C       *                          ELEVATOR DEFLECTION                  *
C       *           NB     (R) = YAWING MOMENT DERIVATIVE DUE TO        *
C       *                          SIDESLIP ANGLE                       *
C       *           NP     (R) = YAWING MOMENT DERIVATIVE DUE TO        *
C       *                          ROLL RATE                            *
C       *           NR     (R) = YAWING MOMENT DERIVATIVE DUE  TO       *
C       *                          YAW RATE                             *
C       *           NDA    (R) = YAWING MOMENT DERIVATIVE DUE TO        *
C       *                          AILERON DEFLECTION                   *
C       *           NDR    (R) = YAWING MOMENT DERIVATIVE DUE TO        *
C       *                          RUDDER DEFLECTION                    *
C       *                                                               *
C       *     ERROR CHECKS                                              *
C       *         (NONE)                                                *
C       *                                                               *
C       *     ASSOCIATED SUBPROGRAMS                                    *
C       *         CALLED BY                                             *
C       *             PINAM                                             *
C       *         CALLS TO                                              *
C       *             (NONE)                                            *
C       *                                                               *
C       *     REFERENCES                                                *
C       *         (NONE)                                                *
C       *                                                               *
C       *     LANGUAGE                                                  *
C       *         STANDARD FORTRAN 77                                   *
C       *****************************************************************

C       ********************** ARGUMENT VARIABLES ********************
      REAL  MACH, ACMASS, IAC(3,3), QBAR, U0

C       ********************** COMMON VARIABLES **********************

C       ***************************************************************
C       *     THE COMMON BLOCK  /SSVAR/  CONTAINS THE STEADY STATE    *
C       * STATE VARIABLES CALCULATED FOR THE GIVEN MANEUVER.          *
C       ***************************************************************
      REAL    P0,Q0,R0
      REAL    PHI0,PSI0,THETA0
      REAL    LIFT
      COMMON /SSVAR/
     1        P0,Q0,R0,
     2        PHI0,PSI0,THETA0,
```

```
      3       LIFT
C         ****************************************************************
C         *     THE COMMON BLOCK /ACGEOM/ CONTAINS AIRCRAFT GEOMETRY     *
C         *  INFORMATION.  INCLUDED IN THIS COMMON BLOCK ARE THE         *
C         *  AIRCRAFT'S WINGSPAN (SPAN), THE AIRCRAFT'S WING AREA        *
C         *  (AREA), AND THE MEAN AERODYNAMIC CHORD (CHORD).             *
C         ****************************************************************
       REAL  SPAN , AREA , CHORD
       COMMON /ACGEOM/
      1             SPAN , AREA , CHORD
C         ****************************************************************
C         *     THE COMMON BLOCK /ADER/ CONTAINS THE GENERALIZED         *
C         *  AERODYNAMIC COEFFICIENT MATRICES.                           *
C         ****************************************************************
       REAL
      1 XU, XA, XDE, YB, YP, YR, YDA, YDR, ZU, ZA, ZDE,
      2 LB, LP, LR, LDA, LDR, MU, MA, MADOT, MQ, MDE,
      3 NB, NP, NR, NDA, NDR
       COMMON /ADER/
      1             XU, XA, XDE, YB, YP, YR, YDA, YDR, ZU, ZA, ZDE,
      2             LB, LP, LR, LDA, LDR, MU, MA, MADOT, MQ, MDE,
      3             NB, NP, NR, NDA, NDR

C         ****************************************************************
C         *    THE COMMON BLOCK  /ADERC/  CONTAINS THE AERODYNAMIC       *
C         * DERIVATIVE COEFFICIENTS NEEDED FOR THE STEADY STATE          *
C         * MANEUVER CALCULATION.                                        *
C         ****************************************************************
       COMMON /ADERC/
      1       CLFTA,CMA,CMDE

C         ****************************************************************
C         *    THE COMMON BLOCK /AERO1/ CONTAINS THE AERODYNAMIC         *
C         *    COEFFICIENT TABLES THAT ARE READ IN FROM THE DATA         *
C         *    FILE.  THE AERODYNAMIC COEFFICIENTS ARE THEN              *
C         *    INTERPOLATED IN THESE TABLES TO OBTAIN INSTANTANEOUS      *
C         *    VALUES.                                                   *
C         ****************************************************************
       REAL
      1   M(13) , QD(8) , A(13) , E(18) , CL2(14) , ACLFT0(13,8) ,
      2   ACLFTA(13,8) , ACLFTE(13) , ACDSB(13) , ACDTRM(13,14) ,
      3   ACYB(13) , ACYDR(13) , ACLB(13) , ACLR(13) , ACLP(13) ,
      4   ACLDR(13) , ACLDA(13) , ACLDSP(13) , ACM0(13) , ACMA(13,13) ,
      5   ACMDE(13) , ACMQ(13) , ACNB(13) , ACNR(13) , ACNP(13) ,
      6   ACNDR(13) , ACNDSP(13) , ACNDA(13) , AKDE(13,18) , NMASUB(8) ,
      7   NMASUP(8) , NMESUB(8) , NMESUP(8) , ANDR(8)
       COMMON /AERO1/
      1   M,QD,A,E,CL2,ACLFT0,ACLFTA,ACLFTE,ACDSB,ACDTRM,ACYB,ACYDR,
      2   ACLB,ACLR,ACLP,ACLDR,ACLDA,ACLDSP,ACM0,ACMA,ACMDE,ACMQ,
      3   ACNB,ACNR,ACNP,ACNDR,ACNDSP,ACNDA,AKDE,NMASUB,NMASUP,
      4   NMESUB,NMESUP,ANDR

C         ********************** LOCAL VARIABLES ***********************
       REAL  PI , LOW , HIGH , DETRIM , P , Q , R , MPRCNT ,
      1      QPRCNT , EPRCNT , APRCNT , LPRCNT , RALPHA , MACHP ,
      2      QBARP , ALPHAP
       REAL CLIFT , CLIFT2 , CLTRM , CLFT0 , CLFTA , CLFTE , CD ,
      1     CDTRM , CDSB , CX , CY , CYB , CYDR , CZ , CL , CLB , CLP ,
      2     CLR , CLDA , CLDR , CLDSP , CM , CM0 , CMA , CMDE , CMQ ,
```

```
     3     CN , CNB , CNP , CNR , CNDA , CNDR , CNDSP , KDE , KDETRM ,
     4     NMA , NME , NDRC
      INTEGER  I , KM , KQ , KE , KA , KL , NM , NQ , NA , NE , NL ,
     1         KM1 , KQ1 , KE1 , KA1 , KL1

C       ********************** DATA STATEMENTS **********************
      DATA  NM /13/
      DATA  NQ /8/
      DATA  NA /13/
      DATA  NE /18/
      DATA  NL /14/
      DATA  MACHP/999.9/
      DATA  QBARP/000.0/
      DATA  ALPHAP/999.9/
      DATA  PI /3.1415927/

C
C       ******************** START EXECUTABLE CODE ******************
C

C   ****************************************************************
C   *                                                              *
C   *       START LINEAR INTERPOLATION PROCEDURE TO DETERMINE      *
C   *             THE NEEDED AERODYNAMIC COEFFICIENTS.             *
C   *                                                              *
C   *  NOTE: IF VALUE NEEDED IS OUT OF THE DATA RANGE, THE SIMULATION*
C   *        WILL CONTINUE WITH THE CLOSEST TABLE VALUE USED.      *
C   ****************************************************************

C       ************************************************************
C       *         FIND LOCATION OF CURRENT MACH NUMBER (MACH)      *

C       *                    IN MACH NUMBER TABLE (M)              *
C       ************************************************************
      IF( ABS(MACH-MACHP) .GT. 0.001)THEN
        IF( MACH .LE. M(1) )THEN
          KM = 1
          MPRCNT = 0.0
        ELSEIF( MACH .GT. M(1)  .AND.  MACH .LT. M(NM) )THEN
          DO 100 I = 2,NM
          IF( MACH .LT. M(I)  .AND.  MACH .GE. M(I-1) )THEN
            KM = I-1
            MPRCNT = ( MACH-M(I-1) ) / ( M(I)-M(I-1) )
          ENDIF
  100     CONTINUE
        ELSE
          KM = NM - 1
          MPRCNT = 1.0
        ENDIF
        KM1 = KM + 1
        MACHP = MACH
      ENDIF

C       ************************************************************
C       *      FIND LOCATION OF CURRENT DYNAMIC PRESSURE (QBAR)    *
C       *                IN DYNAMIC PRESSURE TABLE (QD)            *
C       ************************************************************
      IF( ABS(QBAR-QBARP) .GT. 5.0 ) THEN
        IF( QBAR .LE. QD(1) )THEN
```

```
          KQ = 1
          QPRCNT = 0.0
        ELSEIF( QBAR .GT. QD(1)  .AND.  QBAR .LT. QD(NQ) )THEN
          DO 200 I = 2,NQ
          IF( QBAR .LT. QD(I)  .AND.  QBAR .GE. QD(I-1) )THEN
            KQ = I-1
            QPRCNT = ( QBAR-QD(I-1) ) / ( QD(I)-QD(I-1) )
           ENDIF
  200      CONTINUE
        ELSE
          KQ = NQ - 1
          QPRCNT = 1.0
        ENDIF
        KQ1 = KQ + 1
        QBARP = QBAR
      ENDIF

C        ***************************************************************
C        *    DETERMINATION OF THE ZERO ANGLE OF ATTACK LIFT           *
C        *    COEFFICIENT (CLFT0) AND CHANGE IN LIFT DUE TO ANGLE      *
C        *    OF ATTACK COEFFICIENT (CLFTA) , BY LOCATION OF THE       *
C        *    CURRENT MACH NUMBER (MACH) IN THE MACH NUMBER TABLE (M)  *
C        *    AND THE CURRENT DYNAMIC PRESSURE (QBAR) IN THE DYNAMIC   *
C        *    PRESSURE TABLE (QD).                                     *
C        ***************************************************************
      LOW    = ACLFT0(KM,KQ)+MPRCNT*( ACLFT0(KM1,KQ)-
     1         ACLFT0(KM,KQ) )
      HIGH   = ACLFT0(KM,KQ1)+MPRCNT*( ACLFT0(KM1,KQ1)-
     1         ACLFT0(KM,KQ1) )
      CLFT0 = LOW + QPRCNT*(HIGH-LOW)
      LOW    = ACLFTA(KM,KQ)+MPRCNT*( ACLFTA(KM1,KQ)-
     1         ACLFTA(KM,KQ) )
      HIGH   = ACLFTA(KM,KQ1)+MPRCNT*( ACLFTA(KM1,KQ1)-
     1         ACLFTA(KM,KQ1) )
      CLFTA = LOW + QPRCNT*(HIGH-LOW)
C
C        *      FIND REQUIRED LIFT
C
      ALPHA = (LIFT/QBAR/AREA)/CLFTA      ! alpha is in degrees
C        ***************************************************************
C        *      FIND LOCATION OF CURRENT ANGLE OF ATTACK (ALPHA)       *
C        *                  IN ANGLE OF ATTACK TABLE (A)               *
C        ***************************************************************
      IF( ABS(ALPHA-ALPHAP) .GT. 0.001) THEN
        IF( ALPHA .LE. A(1) )THEN
          KA = 1
          APRCNT = 0.0
        ELSEIF( ALPHA .GT. A(1)  .AND.  ALPHA .LT. A(NA) )THEN
          DO 300 I = 2,NA
          IF( ALPHA .LT. A(I)  .AND.  ALPHA .GE. A(I-1) )THEN
            KA = I-1
            APRCNT = ( ALPHA-A(I-1) ) / ( A(I)-A(I-1) )
          ENDIF
  300     CONTINUE
        ELSE
          KA = NA - 1
          APRCNT = 1.0
        ENDIF
c        ka=4            !prov
        KA1 = KA + 1
```

```
        ALPHAP = ALPHA
      ENDIF

C       ************************************************************
C       *                                                          *
C       *   DETERMINATION OF COEFFICIENTS DEPENDENT ONLY ON MACH   *
C       *   NUMBER BY LOCATION OF CURRENT MACH NUMBER (MACH) IN    *
C       *   MACH TABLE (M).                                        *
C       *                                                          *
C       ************************************************************
        CLFTE =  ACLFTE(KM) + MPRCNT * ( ACLFTE(KM1)-ACLFTE(KM) )
        CYB   =  ACYB(KM)   + MPRCNT * ( ACYB(KM1)-ACYB(KM) )
        CYDR  =  ACYDR(KM)  + MPRCNT * ( ACYDR(KM1)-ACYDR(KM) )
        CLB   =  ACLB(KM )  + MPRCNT * ( ACLB(KM1)-ACLB(KM) )
        CLR   =  ACLR(KM)   + MPRCNT * ( ACLR(KM1)-ACLR(KM) )
        CLP   =  ACLP(KM)   + MPRCNT * ( ACLP(KM1)-ACLP(KM) )
        CLDR  =  ACLDR(KM)  + MPRCNT * ( ACLDR(KM1)-ACLDR(KM) )
        CLDA  =  ACLDA(KM)  + MPRCNT * ( ACLDA(KM1)-ACLDA(KM) )

        CM0   =  ACM0(KM)   + MPRCNT * ( ACM0(KM1)-ACM0(KM) )

        CMDE  =  ACMDE(KM)  + MPRCNT * ( ACMDE(KM1)-ACMDE(KM) )
        CMQ   =  ACMQ(KM)   + MPRCNT * ( ACMQ(KM1)-ACMQ(KM) )
        CNB   =  ACNB(KM)   + MPRCNT * ( ACNB(KM1)-ACNB(KM) )
        CNR   =  ACNR(KM)   + MPRCNT * ( ACNR(KM1)-ACNR(KM) )
        CNP   =  ACNP(KM)   + MPRCNT * ( ACNP(KM1)-ACNP(KM) )
        CNDR  =  ACNDR(KM)  + MPRCNT * ( ACNDR(KM1)-ACNDR(KM) )
        CNDA  =  ACNDA(KM)  + MPRCNT * ( ACNDA(KM1)-ACNDA(KM) )
C       ************************************************************
C       *  DETERMINATION OF PITCHING MOMENT DUE TO ANGLE OF ATTACK  *
C       *  COEFFICIENT (CMA) WHICH IS DEPENDENT ON BOTH THE MACH    *
C       *  NUMBER AND ANGLE OF ATTACK BY LOCATION OF CURRENT MACH   *
C       *  NUMBER (MACH) IN MACH NUMBER TABLE (M) AND CURRENT       *
C       *  ANGLE OF ATTACK (ALPHA) IN ANGLE OF ATTACK TABLE (A).    *
C       ************************************************************
      CMA  = (ACMA(KM,KA)+ACMA(KM1,KA)-ACMA(KM,KA1)-ACMA(KM1,KA1))/
     1         (2*(A(KA)-A(KA1)))*57.3

C
C       *  FIND REQUIRED ELEVATOR DEFLECTION
C
      DELEV = -(CMA*ALPHA/57.3)/CMDE

C       ************************************************************
C       *   FIND LOCATION OF CURRENT ELEVATOR DEFLECTION (DELEV)   *
C       *              IN ELEVATOR DEFLECTION TABLE (E)            *
C       ************************************************************
      IF( DELEV .EQ. E(1) )THEN
        KE = 1
        EPRCNT = 0.0
      ELSEIF( DELEV .GT. E(1)  .AND.  DELEV .LT. E(NE) )THEN
        DO 400 I = 2,NE
        IF( DELEV .LT. E(I)  .AND.  DELEV .GE. E(I-1) )THEN
            KE = I-1
            EPRCNT = ( DELEV-E(I-1) ) / ( E(I)-E(I-1) )
        ENDIF
  400   CONTINUE
      ELSE
        KE = NE - 1
        EPRCNT = 1.0
```

```
      ENDIF
      KE1 = KE + 1

C        ***************************************************************
C        *  DETERMINATION OF AEROELASTIC RATIOS WHICH ARE DEPENDENT    *
C        *  ON DYNAMIC PRESSURE BY INTERPOLATION IN APPROPRIATE        *
C        *  TABLES.                                                    *
C        *   (THESE ARE CORRECTION TERMS PRESENT IN AVAILABLE DATA     *
C        *   DUE TO NON-LINEARILTIES ATTRIBUTED TO DYNAMIC PRESSURE)   *
C        ***************************************************************
      IF( MACH .LE. 0.9 )THEN
        NMA  =  NMASUB(KQ)  + QPRCNT * ( NMASUB(KQ1)-NMASUB(KQ) )
        NME  =  NMESUB(KQ)  + QPRCNT * ( NMESUB(KQ1)-NMESUB(KQ) )
        NDRC =  ANDR(KQ)    + QPRCNT * ( ANDR(KQ1)-ANDR(KQ) )
      ELSE
        NMA  =  NMASUP(KQ)  + QPRCNT * ( NMASUP(KQ1)-NMASUP(KQ) )
        NME  =  NMESUP(KQ)  + QPRCNT * ( NMESUP(KQ1)-NMESUP(KQ) )
        NDRC =  ANDR(KQ)    + QPRCNT * ( ANDR(KQ1)-ANDR(KQ) )
      ENDIF

C        ***************************************************************
C        *  DETERMINATION OF NON-LINEAR CORRECTION FACTOR (KDE)        *
C        *  WHICH IS DEPENDENT ON BOTH MACH NUMBER AND ELEVATOR        *
C        *  DEFLECTION BY LOCATION OF CURRENT MACH NUMBER (MACH)       *
C        *  IN THE MACH NUMBER TABLE (M) AND THE CURRENT ELEVATOR      *
C        *  DEFLECTION (DELEV) IN THE ELEVATOR DEFLECTION TABLE (E).   *
C        *   ( THIS COEFFICIENT IS A CORRECTION TERM PRESENT IN        *
C        *     AVAILABLE DATA DUE TO NON-LINEARITIES ATTRIBUTED        *
C        *     TO MACH NUMBER AND ELEVATOR DEFLECTION )                *
C        ***************************************************************
        LOW  = AKDE(KM,KE)+MPRCNT*(AKDE(KM1,KE)-AKDE(KM,KE))
        HIGH = AKDE(KM,KE1)+MPRCNT*(AKDE(KM1,KE1)-AKDE(KM,KE1))
        KDE = LOW + EPRCNT*(HIGH-LOW)

C        ***************************************************************
C        *      CALCULATE THE LIFT COEFFICIENT AND THE ELEVATOR        *
C        *           DEFLECTION FOR TRIM FLIGHT CONDITIONS.            *
C        ***************************************************************
      CLTRM = CLFT0 + CLFTA*ALPHA
      if (cltrm.gt.1)print*,'Warning: lift coefficient too high',cltrm
      DETRIM = ( -(CM0+CMA*NMA*ALPHA/57.3) )/(CMDE*NME)   !CHECK

C        ***************************************************************
C        *   FIND LOCATION OF TRIM ELEVATOR DEFLECTION (DELEV)         *
C        *             IN ELEVATOR DEFLECTION TABLE (E)                *
C        ***************************************************************
      IF( DETRIM .EQ. E(1) )THEN
        KE = 1
        EPRCNT = 0.0
      ELSEIF( DETRIM .GT. E(1)  .AND.  DETRIM .LT. E(NE) )THEN
        DO 500 I = 2,NE
        IF( DETRIM .LT. E(I)  .AND.  DETRIM .GE. E(I-1) )THEN
            KE = I-1
            EPRCNT = ( DETRIM-E(I-1) ) / ( E(I)-E(I-1) )
        ENDIF
  500   CONTINUE
      ELSE
        KE = NE - 1
        EPRCNT = 1.0
      ENDIF
```

```
      KE1 = KE + 1

C        ************************************************************
C        *  DETERMINATION OF NON-LINEAR CORRECTION FACTOR (KDETRM)   *
C        *  WHICH IS DEPENDENT ON BOTH MACH NUMBER AND ELEVATOR      *
C        *  DEFLECTION BY LOCATION OF CURRENT MACH NUMBER (MACH)     *
C        *  IN THE MACH NUMBER TABLE (M) AND THE TRIM ELEVATOR       *
C        *  DEFLECTION (DETRIM) IN THE ELEVATOR DEFLECTION TABLE (E).*
C        *   ( THIS COEFFICIENT IS A CORRECTION TERM PRESENT IN      *
C        *     AVAILABLE DATA DUE TO NON-LINEARITIES ATTRIBUTED      *
C        *     TO MACH NUMBER AND ELEVATOR DEFLECTION )              *
C        ************************************************************
      LOW  = AKDE(KM,KE)+MPRCNT*(AKDE(KM1,KE)-AKDE(KM,KE))
      HIGH = AKDE(KM,KE1)+MPRCNT*(AKDE(KM1,KE1)-AKDE(KM,KE1))
      KDETRM = LOW + EPRCNT*(HIGH-LOW)

C        ************************************************************
C        *      CALCULATE THE TOTAL LIFT COEFFICIENT (CLIFT) AND    *
C        *            LIFT COEFFICIENT SQUARED (CLIFT2)             *
C        ************************************************************
      CLIFT = CLTRM + CLFTE * ( DELEV*KDE - DETRIM*KDETRM )
      CLIFT2 = CLIFT * CLIFT

C        ************************************************************
C        *    INTERPOLATE FOR THE DRAG COEFFICIENT FOR TRIM FLIGHT   *
C        *    CONDITIONS (CDTRM) AS A FUNCTION OF MACH NUMBER AND    *
C        *    LIFT COEFFICIENT SQUARED.                              *
C        ************************************************************
      IF( CLIFT2 .EQ. CL2(1) )THEN
        KL = 1
        LPRCNT = 0.0
      ELSEIF( CLIFT2 .GT. CL2(1)  .AND.  CLIFT2 .LT. CL2(NL) )THEN
        DO 600 I = 2,NL
        IF( CLIFT2 .LT. CL2(I)  .AND.  CLIFT2 .GE. CL2(I-1) )THEN
            KL = I-1
            LPRCNT = ( CLIFT2-CL2(I-1) ) / ( CL2(I)-CL2(I-1) )
        ENDIF
  600   CONTINUE
      ELSE
        KL = NL - 1
        LPRCNT = 1.0
      ENDIF
        KL1 = KL + 1

      LOW   = ACDTRM(KM,KL)+MPRCNT*( ACDTRM(KM1,KL)-
     1      ACDTRM(KM,KL) )
      HIGH  = ACDTRM(KM,KL1)+MPRCNT*( ACDTRM(KM1,KL1)-
     1        ACDTRM(KM,KL1) )
      CD = LOW + LPRCNT*(HIGH-LOW)

C        ************************************************************
C        *  ESTIMATION OF OTHER STABILITY DERIVATIVES: CDU, CDA, CLU,*
C        *  CMU, CMADOT.                                             *
C        ************************************************************

      CDMACH = (ACDTRM(KM,KL)+ACDTRM(KM,KL1)
     1        -ACDTRM(KM1,KL1)-ACDTRM(KM1,KL))/(2*(M(KM)-M(KM1)))
      CDU = MACH*CDMACH

      CDCL2 = (ACDTRM(KM,KL)-ACDTRM(KM,KL1)
```

```
     1          -ACDTRM(KM1,KL1)+ACDTRM(KM1,KL))/
     2           (2*(CL2(KL)-CL2(KL1)))
      CDA = 2*CLIFT*CLFTA*CDCL2

      CLU = MACH**2/(1-MACH**2)*CLIFT

      CMMACH = (ACMA(KM,KA)+ACMA(KM,KA1)
     1          -ACMA(KM1,KA)-ACMA(KM1,KA1))/(2*(M(KM)-M(KM1)))
      CMU = MACH*CMMACH

      EPSA   = 0.465              ! EPSA= DEPS/DALPHA FUNCTION OF GEOM.
      CMADOT = CMQ * EPSA

C     ****************************************************************
C     *  DERIVATIVES WHICH ARE SMALL AND CAN BE NEGLECTED.           *
C     ****************************************************************
      CXDE = 0.
      CYP  = 0.
      CYR  = 0.
      CYDA = 0.

C     ****************************************************************
C     *  WRITING THE FORCE AND MOMENT COEFFICIENTS TO THE MATRICES*
C     *  OF THE COMMON BLOCK ADER.                                   *
C     ****************************************************************

      C1 = QBAR*AREA/ACMASS
      C2 = QBAR*AREA*SPAN/IAC(1,1)
      C3 = QBAR*AREA*CHORD/IAC(2,2)
      C4 = QBAR*AREA*SPAN/IAC(3,3)
      C5 = SPAN/(2*U0)
      C6 = CHORD/(2*U0)
      DTOR=PI/180.
      CLFTA=CLFTA/DTOR
      XU  = -C1 * (CDU + 2*CD)/U0
      XA  =  C1 * (-CDA + CL)
      XDE =  C1 * CXDE / DTOR
      YB  =  C1 * CYB / DTOR
      YP  =  C1 * CYP * C5 /DTOR
      YR  =  C1 * CYR * C5 /DTOR
      YDA =  C1 * CYDA /DTOR
      YDR =  C1 * CYDR /DTOR
      ZU  = -C1 * (CLU + 2*CL)/U0
      ZA  = -C1 * (CLFTA + CD)
      ZDE = -C1 * CLFTE / DTOR

      LB  = C2 * CLB / DTOR
      LP  = C2 * CLP * C5 / DTOR
      LR  = C2 * CLR * C5 / DTOR
      LDA = C2 * CLDA / DTOR
      LDR = C2 * CLDR / DTOR
      MU  = C3 * (CMU + 2*CM) /U0
      MA  = C3 * CMA
      MADOT = C3 * CMADOT *C6 / DTOR
      MQ  = C3 * CMQ *C6 / DTOR
      MDE = C3 * CMDE / DTOR
      NB  = C4 * CNB / DTOR
      NP  = C4 * CNP * C5 / DTOR
      NR  = C4 * CNR * C5 / DTOR
      NDA = C4 * CNDA / DTOR
```

```
      NDR = C4 * CNDR  / DTOR
!      WRITE(*,*)XU,XA,XDE,YB,YP,YR,YDA,YDR,ZU,ZA,ZDE,
!     1          LB,LP,LR,LDA,LDR,MU,MA,MADOT,MQ,MDE,NB,NP,NR,NDA,NDR
      RETURN
      END
```

```
      SUBROUTINE PIMAM
     G           (U0,IAC,
     Y            A)
C    ****************************************************************
C    *                                                 SEPT 11, 1991  *
C    *                                                                *
C    *     CALLING SEQUENCE                                           *
C    *         SUBROUTINE PIMAM(IAC,A)                                *
C    *                                                                *
C    *     DESCRIPTION : A-MATRIX                                     *
C    *                   --------                                     *
C    *                                                                *
C    *                                                                *
C    *     RESTRICTIONS                                               *
C    *         (NONE)                                                 *
C    *                                                                *
C    *     COMMUNICATION                                              *
C    *         ARGUMENT LIST                                          *
C    *             GIVEN                                              *
C    *               A                                                *
C    *             YIELDED                                            *
C    *               A                                                *
C    *                                                                *
C    *     LOCAL VARIABLES                                            *
C    *             SPAN    (R) = AIRCRAFT WINGSPAN TO WHICH THE       *
C    *                           AERODYNAMIC DATA IS REFERENCED (ft)  *
C    *             CHORD   (R) = AIRCRAFT MEAN AERODYNAMIC CHORD      *
C    *                           LENGTH TO WHICH THE AERODYNAMIC      *
C    *                           DATA IS REFERENCED (ft)              *
C    *             AREA    (R) = AIRCRAFT WING AREA TO WHICH THE      *
C    *                           AERODYNAMIC DATA IS REFERENCED       *
C    *                           (ft^2)                               *
C    *                                                                *
C    *     COMMON VARIABLES                                           *
C    *             P0      (R) = AIRCRAFT ROLL RATE WRITTEN IN THE    *
C    *                           AIRCRAFT FRAME (rad/sec)             *
C    *             Q0      (R) = AIRCRAFT PITCH RATE WRITTEN IN THE   *
C    *                           AIRCRAFT FRAME (rad/sec)             *
C    *             R0      (R) = AIRCRAFT YAW RATE WRITTEN IN THE     *
C    *                           AIRCRAFT FRAME (rad/sec)             *
C    *             PHI0    (R) = SS BANK ANGLE                        *
C    *             PSI0    (R) = SS YAW ANGLE                         *
C    *             THETA0  (R) = SS PITCH ANGLE                       *
C    *             OMEGA   (R) = AIRCRAFT ANGULAR VELOCITY VECTOR     *
C    *                           WRITTEN IN THE AIRCRAFT FRAME        *
C    *                           (rad/sec)                            *
C    *             DELEV   (R) = SYMMETRICAL DEFLECTION OF THE        *
C    *                           HORIZONTAL TAIL SURFACE (deg)        *
C    *             DRUD    (R) = RUDDER DEFLECTION (deg)              *
C    *             DAIL    (R) = ASYMMETRIC DEFLECTION OF THE         *
C    *                           HORIZONTAL TAIL SURFACE (deg)        *
C    *             XU      (R) = X-FORCE DERIVATIVE DUE TO VELOCITY   *
C    *             XA      (R) = X-FORCE DERIVATIVE DUE TO ANGLE OF   *
C    *                           ATTACK                               *
C    *             YB      (R) = SIDE FORCE DERIVATIVE DUE TO SIDE-   *
C    *                           SLIP ANGLE                           *
C    *             YDR     (R) = SIDE FORCE DERIVATIVE DUE TO RUDDER  *
C    *                           DEFLECTION                           *
C    *             ZU      (R) = Z-FORCE DERIVATIVE DUE TO VELOCITY   *
C    *             ZA      (R) = Z-FORCE DERIVATIVE DUE TO ANGLE OF   *
```

```
C     *                                                                *
C     *                        ATTACK                                  *
C     *          ZDE     (R) = Z-FORCE DERIVATIVE DUE TO ELEVATOR      *
C     *                        DEFLECTION                              *
C     *          LB      (R) = ROLLING MOMENT DERIVATIVE DUE TO        *
C     *                        SIDESLIP ANGLE                          *
C     *          LP      (R) = ROLLING MOMENT DERIVATIVE DUE TO        *
C     *                        ROLL RATE                               *
C     *          LR      (R) = ROLLING MOMENT DERIVATIVE DUE TO        *
C     *                        YAW RATE                                *
C     *          LDR     (R) = ROLLING MOMENT DERIVATIVE DUE TO        *
C     *                        RUDDER DEFLECTION                       *
C     *          LDA     (R) = ROLLING MOMENT DERIVATIVE DUE TO        *
C     *                        AILERON DEFLECTION                      *
C     *          MU      (R) = PITCHING MOMENT DERIVATIVE DUE TO       *
C     *                        VELOCITY                                *
C     *          MA      (R) = PITCHING MOMENT DERIVATIVE DUE TO       *
C     *                        ANGLE OF ATTACK                         *
C     *          MADOT   (R) = PITCHING MOMENT DERIVATIVE DUE TO       *
C     *                        ANGLE OF ATTACK CHANGE RATE             *
C     *          MQ      (R) = PITCHING MOMENT DERIVATIVE DUE TO       *
C     *                        PITCH RATE                              *
C     *          MDE     (R) = PITCHING MOMENT DERIVATIVE DUE TO       *
C     *                        ELEVATOR DEFLECTION                     *
C     *          NB      (R) = YAWING MOMENT DERIVATIVE DUE TO         *
C     *                        SIDESLIP ANGLE                          *
C     *          NP      (R) = YAWING MOMENT DERIVATIVE DUE TO         *
C     *                        ROLL RATE                               *
C     *          NR      (R) = YAWING MOMENT DERIVATIVE DUE  TO        *
C     *                        YAW RATE                                *
C     *          NDA     (R) = YAWING MOMENT DERIVATIVE DUE TO         *
C     *                        AILERON DEFLECTION                      *
C     *          NDR     (R) = YAWING MOMENT DERIVATIVE DUE TO         *
C     *                        RUDDER DEFLECTION                       *
C     *                                                                *
C     *    ERROR CHECKS                                                *
C     *        (NONE)                                                  *
C     *                                                                *
C     *    ASSOCIATED SUBPROGRAMS                                      *
C     *        CALLED BY                                               *
C     *            PLTMOD                                              *
C     *        CALLS TO                                                *
C     *            PMADER                                              *
C     *                                                                *
C     *    REFERENCES                                                  *
C     *        (NONE)                                                  *
C     *                                                                *
C     *    LANGUAGE                                                    *
C     *        STANDARD FORTRAN 77                                     *
C     ******************************************************************

      INTEGER NN
      PARAMETER (NN=16)

C        ********************** ARGUMENT VARIABLES ******************
      REAL A(NN,NN),IAC(3,3)

C        ********************** COMMON VARIABLES ********************

C        ************************************************************
C        *     THE COMMON BLOCK /ACGEOM/ CONTAINS AIRCRAFT GEOMETRY  *
```

```
C        *  INFORMATION.  INCLUDED IN THIS COMMON BLOCK ARE THE      *
C        *  AIRCRAFT'S WINGSPAN (SPAN), THE AIRCRAFT'S WING AREA     *
C        *  (AREA), AND THE MEAN AERODYNAMIC CHORD (CHORD).          *
C        ************************************************************
      REAL  SPAN , AREA , CHORD
      COMMON /ACGEOM/
     1            SPAN , AREA , CHORD
C        ************************************************************
C        *     THE COMMON BLOCK /ADER/ CONTAINS THE GENERALIZED      *
C        *  AERODYNAMIC COEFFICIENT MATRICES.                        *
C        ************************************************************
      REAL
     1 XU, XA, XDE, YB, YP, YR, YDA,YDR, ZU, ZA, ZDE,
     2 LB, LP, LR, LDA, LDR, MU, MA, MADOT, MQ, MDE,
     3 NB, NP, NR, NDA, NDR
      COMMON /ADER/
     1            XU, XA, XDE, YB, YP, YR, YDA, YDR, ZU, ZA, ZDE,
     2            LB, LP, LR, LDA, LDR, MU, MA, MADOT, MQ, MDE,
     3            NB, NP, NR, NDA, NDR

C        ************************************************************
C        *    THE COMMON BLOCK  /SSVAR/  CONTAINS THE STEADY STATE  *
C        * STATE VARIABLES CALCULATED FOR THE GIVEN MANEUVER.       *
C        ************************************************************
      REAL   P0,Q0,R0
      REAL   PHI0,PSI0,THETA0
      REAL   DELEV,DAIL,DRUD
      REAL   ALPHA
      COMMON /SSVAR/
     1       P0,Q0,R0,
     2       PHI0,PSI0,THETA0,
     3       DELEV,DAIL,DRUD,
     4       ALPHA
C        ********************** LOCAL VARIABLES **********************
      REAL  G

C        ********************** DATA STATEMENTS **********************
      DATA  G /32.2/            ! CHECK IF UNIT IS OK

C
C        ******************** START EXECUTABLE CODE ******************
C
      A(1,1)  =   XU
      A(1,2)  =   R0
      A(1,3)  =   XA/U0-Q0
      A(1,7)  =   XDE/U0
      A(1,11) = - G/U0 * COS(THETA0)
      A(1,12) = - XU/U0
      A(1,15) = - XA/U0**2

      A(2,1)  = - R0
      A(2,2)  =   YB/U0
      A(2,3)  =   P0
      A(2,4)  =   YP/U0
      A(2,6)  =   YR/U0 -1
      A(2,8)  =   YDA/U0
      A(2,9)  =   YDR/U0
      A(2,10) =   G/U0 * COS(THETA0) * COS(PHI0)
      A(2,11) = - G/U0 * SIN(THETA0) * SIN(PHI0)
      A(2,13) = - YB/U0**2
```

```
      A(3,1)  =   ZU + Q0
      A(3,2)  = - P0
      A(3,3)  =   ZA/U0
      A(3,5)  =   1.
      A(3,7)  =   ZDE/U0
      A(3,10) = - G/U0 * COS(THETA0) * SIN(PHI0)
      A(3,11) = - G/U0 * SIN(THETAO) * COS(PHI0)
      A(3,12) = - ZU/U0
      A(3,15) = - ZA/U0**2

      A(4,2)  =   LB
      A(4,4)  =   LP + (Q0*IAC(1,3) - R0*IAC(1,2))/IAC(1,1)
      A(4,5)  =  (2*Q0*IAC(2,3) + P0*IAC(1,3) -
     1             R0*(IAC(3,3)-IAC(2,2)))/IAC(1,1)
      A(4,6)  =   LR - (2*R0*IAC(2,3) + P0*IAC(1,2) +
     1             Q0*(IAC(3,3)-IAC(2,2)))/IAC(1,1)
      A(4,8)  =   LDA
      A(4,9)  =   LDR
      A(4,13) = - LB/U0

      A(5,1)  =   U0*MU + MADOT*(Q0+ZU)
      A(5,2)  =  -MADOT*PO
      A(5,3)  =   MA + MADOT*ZA/U0
      A(5,4)  = -(2*P0*IAC(1,3) + Q0*IAC(2,3) +
     1             R0*(IAC(1,1)-IAC(3,3)))/IAC(2,2)
      A(5,5)  =   MADOT + MQ + (R0*IAC(1,2)-P0*IAC(2,3))/IAC(2,2)
      A(5,6)  =  (2*R0*IAC(1,3) + Q0*IAC(1,2) -
     1             P0*(IAC(1,1)-IAC(3,3)))/IAC(2,2)
      A(5,7)  =   MDE + MADOT*ZDE/U0
      A(5,10) = - MADOT*G/U0 * COS(THETA0) * SIN(PHI0)     ! CHECK
      A(5,11) = - MADOT*G/U0 * SIN(THETA0) * COS(PHI0)
      A(5,12) = - (MU + MADOT*ZU/U0)
      A(5,15) = - (MADOT*ZA+MA)/U0

      A(6,2)  =   NB
      A(6,4)  =   NP + (2*P0*IAC(1,2)-Q0*(IAC(2,2)-IAC(1,1)))/IAC(3,3)
      A(6,5)  = -(2*Q0*IAC(1,2) + R0*IAC(1,3) +
     1             P0*(IAC(2,2)-IAC(1,1)))/IAC(3,3)
      A(6,6)  =   NR - (Q0*IAC(1,3)-P0*IAC(2,3))/IAC(3,3)
      A(6,8)  =   NDA
      A(6,9)  =   NDR
      A(6,13) = - NB/U0


      A(10,4) =  1.0
      A(11,5) =  COS(PHI0)


      RETURN
      END
```

```
      SUBROUTINE RICSOL
     G             ( A , B , Qstar , R ,
     Y               K , ERRFLG )

C     ****************************************************************
C     *                                                              *
C     *    CALLING SEQUENCE                                          *
C     *        SUBROUTINE RICSOL                                     *
C     *     G             ( A , B , Qstar , R ,                      *
C     *     Y               K , ERRFLG )                             *
C     *                                                              *
C     *    DESCRIPTION : LINEAR OPTIMAL QUADRATIC TRACKER            *
C     *                                                              *
C     *    THIS SUBROUTINE GENERATES THE OPTIMAL FEEDBACK GAIN       *
C     *    MATRICES FOPT AND GOPT ASSOCIATED WITH THE OPTIMAL        *
C     *    CONTROL u(t) GIVEN BY:                                    *
C     *                                                              *
C     *     u(t) = FOPT*x(t) + GOPT*Yc(t)                            *
C     *                                                              *
C     *    GIVEN THE LINEAR TIME INVARIANT SYSTEM                    *
C     *                                                              *
C     *      .                                                       *
C     *      X(t) = Ax(t) + Bu(t)       x(0)=x (t)                   *
C     *                                      0                       *
C     *                                                              *
C     *    WITH OUTPUT                                               *
C     *                                                              *
C     *     Y(t) = Cx(t)                                             *
C     *                                                              *
C     *    THE OPTIMAL CONTROL WHICH MINIMIZES                       *
C     *                                                              *
C     *                      inf          T              T          *
C     *      J(u) = INTEGRAL | [(Yc-Y)  Q(Yc-Y) + u Ru]dt             *
C     *                      0                                       *
C     *                                                              *
C     *     IS GIVEN BY                                              *
C     *                  -1 T                                        *
C     *       u(t) = -R  B [Kx(t) - s(t)]                            *
C     *                                                              *
C     *     WITH                                                     *
C     *                -1 T                                          *
C     *     FOPT = -R  B K                                           *
C     *                                                              *
C     *                -1 T  T     -1 T  -1 T                       *
C     *     GOPT = -R  B [A - BR  B K]  C  QC                        *
C     *                                                              *
C     *     WHERE K AND s SATISFY:                                   *
C     *                T         -1 T      T                         *
C     *        KA + A K - KBR  B K - C QC = 0                        *
C     *                                                              *
C     *              T   -1 T        T                               *
C     *        [A -BR  B K]s + C QYc = 0                             *
C     *                                                              *
C     *     AND Yc(t) IS THE DESIRED (COMMANDED) OUTPUT VECTOR       *
C     *                                                              *
C     *        AIRCRAFT (A,B) PAIR MUST BE CONTROLLABLE              *
C     *    WEIGHTING Q AND R MUST BE POSITIVE, SEMIDEFINITE AND      *
C     *    DEFINITE RESPECTIVELY.                                    *
C     *                                                              *
C     *    COMMUNICATION                                             *
C     *        ARGUMENT LIST                                         *
```

```
C        *                                                               *
C        *            GIVEN                                              *
C        *                                                               *
C        *                A       (R) = PLANT MATRIX                     *
C        *                B       (R) = CONTROL INPUT MATRIX             *
C        *                QSTAR   (R) = STATE WEIGHTING MATRIX           *
C        *                R       (R) = INPUT WEIGHTING MATRIX           *
C        *                              FEEDFORWARD MATRIX               *
C        *                K       (R) = SOLUTION OF THE RICCATI          *
C        *                              EQUATION                         *
C        *                ERRFLG  (L) = ERROR FLAG                       *
C        *                                                               *
C        *    ERRORS                                                     *
C        *        (NONE)                                                 *
C        *                                                               *
C        *    ASSOCIATED SUBPROGRAMS                                     *
C        *        CALLED BY                                              *
C        *            PLTMOD                                             *
C        *        CALLS TO                                               *
C        *            TRNPOS                                             *
C        *            MATMUL                                             *
C        *            RICCND                                             *
C        *                                                               *
C        *    LANGUAGE                                                   *
C        *        STANDARD FORTRAN 77                                    *
C        *****************************************************************

      PARAMETER (NN = 16,NR = 3,NM = 3)
C
C     ************** ARGUMENT VARIABLES ********************
C

      REAL A(NN,NN), B(NN,NR), R(NR,NR),
     &     ATRAN(NN,NN),BTRAN(NR,NN),
     &     K(NN,NN),TNR(NN,NR),rinv(nr,nr),
     &     TNM(NN,NM),BRB(NN,NN),QSTAR(NN,NN)

C
C     ************** LOCAL VARIABLES ***********************
C

      DOUBLE PRECISION AD(NN,NN), BRBD(NN,NN),QSTARD(NN,NN)
      DOUBLE PRECISION Z(2*NN,2*NN),W(2*NN,2*NN),ER(2*NN),
     &       EI(2*NN),WORK(NN),SCALE(2*NN)
      INTEGER ITYPE(2*NN),IPVL(2*NN),IPVS(NN)

      LOGICAL IER, ERRFLG
C
C     **** CONSTANT MATRICES ****
C
      CALL TRNPOS(A,NN,NN,ATRAN)
      CALL TRNPOS(B,NN,NR,BTRAN)
      DO 70 I = 1,NR
   70 RINV(I,I) = 1.0/R(I,I)
      CALL MATMUL(NN,NR,B,NR,NR,RINV,NN,NR,TNR,IER)
        IF (IER) GOTO 500
      CALL MATMUL(NN,NR,TNR,NR,NN,BTRAN,NN,NN,BRB,IER)
        IF (IER) GOTO 500

C
C    **** SET UP DOUBLE MATRICES
```

```
C
      DO I=1,NN
        DO J=1,NN
          AD(I,J)=A(I,J)
          QSTARD(I,J)=QSTAR(I,J)
          BRBD(I,J)=BRB(I,J)
        ENDDO
      ENDDO
C
C     *** SOLVE RICCATI EQUATION ***
C
      CALL RICCND(NN,NN,NN,2*NN,NN,2*NN,AD,BRBD,QSTARD,Z,W,
     1            ER,EI,WORK,SCALE,ITYPE,IPVL,IPVS)

      write(21,'(1x,''er'',16f8.3)')(er(i),i=1,16)
      write(21,'(1x,''ei'',16f8.3)')(ei(i),i=1,16)
      DO I=1,NN
        DO J=1,NN
          K(I,J)=QSTARD(I,J)
        ENDDO
      ENDDO
C
      RETURN

C        ****** ERROR ABORT ******
  500 WRITE(1,*)' GAIN:  ERROR IN MATMUL'
      ERRFLG = .TRUE.
      RETURN
      END
```

```
      SUBROUTINE INTEGRAL(A,N,N11,W,T,RES)
      PARAMETER(NN=16)
      REAL A(nN,NN),W(NN,NN),QQ(NN,NN),RES(NN,NN),EWE(NN,NN),
     &     EXX(NN,NN)
      INTEGER I,II,J,JJ,L,K,P,Q
      INTEGER IPVT(NN),NEVA(NN)
      REAL A11(NN,NN),A22(NN,NN),A12(NN,NN),JO(NN),
     &     TMNN(NN,NN),TM(NN,NN)
      REAL ASUB(NN,NN),WR(NN),WI(NN),ZR(NN,NN),ZI(NN,NN)
      COMPLEX DIAG(NN),OFDIAG(NN)
      COMPLEX REVA(NN,NN),EIGVA(NN),CTMNN1(NN,NN),CTMNN(NN,NN),
     &        SLAMBDA,JD(NN,NN),WTILD(NN,NN),Z(NN),WC(NN,NN),
     &        JAY,ALPHA,WIJ,
     &        Y(NN,NN),F(NN,NN),CEXX(NN,NN)
      LOGICAL ERRFLG
      DATA JAY/(0.,1.)/

C     ****************************************************************
C     * PARTITION MATRIX A                                           *
C     ****************************************************************
      M22=N-N11
      DO 200 I=1,N11
        DO 200 J=1,N11
  200   A11(J,I)=A(J,I)
      DO 210 I=N11+1,N
        DO 210 J=N11+1,N
  210   A22(J-N11,I-N11)=A(J,I)
      DO 220 I=1,N11
        DO 220 J=N11+1,N
  220   A12(I,J-N11)=A(I,J)

C     ****************************************************************
C     *  CALCULATE EIGENSTRUCTURE OF MATRIX A11                      *
C     ****************************************************************

      DO 330 I = 1,N11
        DO 330 J = 1,N11
          ctmnn1(i,j)=a11(i,j)
  330     ASUB(I,J) = A11(I,J)
      CALL EIGV(1,ASUB,NN,N11,WR,WI,ZR,ZI,IERR)
          IF(IERR.NE.0) GOTO 6000
!      WRITE(1,1009)
       write(21,'(1x,''erol'',16f8.3)')(wr(i),i=1,16)
       write(21,'(1x,''eiol'',16f8.3)')(wi(i),i=1,16)
      DO 360 J=1,N11
        SUM = 0.0
        DO 340 I=1,N11
  340     SUM = SUM + ZR(I,J)*ZR(I,J) + ZI(I,J)*ZI(I,J)
        ZMAG = SQRT(SUM)
        IF(ZMAG.LE..001) ZMAG=1.0
        DO 350 I=1,NN
          ZR(I,J) = ZR(I,J)/ZMAG
          ZI(I,J) = ZI(I,J)/ZMAG
          REVA(I,J)=CMPLX(ZR(I,J),ZI(I,J))
  350   CONTINUE
        EIGVA(J)=CMPLX(WR(J),WI(J))
  360 CONTINUE
!      DO 370 J=1,N11
!       WRITE(6,1007) WR(J),WI(J),(ZR(I,J),I=1,N11)
!  370   WRITE(6,1014) (ZI(I,J),I=1,N11)
```

```
c-----prov eigenvector check
       call cMATMULD(n11,n11,NN,ctmnn1,n11,n11,NN,reva,n11,n11,NN,
     1                 ctmnn,errflg)
       do j=1,n11
         do i=1,n11
           ctmnn1(i,j)=reva(i,j)*eigva(j)
           ctmnn(i,j)=ctmnn(i,j)-ctmnn1(i,j)
         enddo
       enddo
C      ****************************************************************
C      *  JORDAN FORM OF MATRIX A22 :                                 *
C      *                             -1                               *
C      *                   JO = Q  *A*Q                              *
C      ****************************************************************
       ! TEST IF MATRIX HAS RIGHT FORM
       EIGVA(N11+1)=A22(1,1)
       EIGVA(N11+2)=A22(3,3)/2
       EIGVA(N11+3)=A22(3,3)/2
       EIGVA(N11+4)=A22(5,5)/2
       EIGVA(N11+5)=A22(5,5)/2
       JO(N11+2)=1
       JO(N11+4)=1

       QQ(1,1)=1.
       QQ(2,2)=A22(3,3)/2
       QQ(3,3)=4./A22(3,3)
       QQ(2,3)=1.
       QQ(3,2)=1.
       QQ(4,4)=A22(5,5)/2
       QQ(5,5)=4./A22(5,5)
       QQ(4,5)=1.
       QQ(5,4)=1.

C      ****************************************************************
C      * CALCULATE F WHERE P*F = -A12*Q                               *
C      ****************************************************************
       DO 245 I=1,n11
         DO 245 J=1,m22
 245     A12(I,J)=-A12(I,J)
       CALL MATMULD(N11,M22,NN,A12,M22,M22,NN,QQ,N11,M22,NN,TMNN,
     1              ERRFLG)
       DO 250 I=1,N11
         DO 250 J=1,N11
           F(J,I)=TMNN(J,I)
 250       CTMNN(J,I)=REVA(J,I)
       CALL CGECO(CTMNN,NN,N11,IPVT,RCOND,Z)
       IF (RCOND+1.0 .EQ. 1.0) THEN ERRFLG=.TRUE.
       DO 260 J=1,M22
         CALL CGESL(CTMNN,NN,N11,IPVT,F(1,J),0)
  260  CONTINUE

C      ****************************************************************
C      *  SOLVE LAMBDA*Y-Y*JD = F                                     *
C      ****************************************************************
       DO 270 I=1,N11
         Y(I,1)=F(I,1)/(EIGVA(I)-EIGVA(N11+1))   !COMPLEX DIVISION
         DO 270 J=2,M22
           Y(I,J)=(F(I,J)+Y(I,J-1)*JO(J-1))/(EIGVA(I)-EIGVA(N11+J))
  270  CONTINUE
```

```
C     ****************************************************************
C     * CALC X=P*Y AND SET UP NEW UNITARY MATRIX P                   *
C     ****************************************************************
      CALL CMATMULD(N11,N11,NN,REVA,N11,M22,NN,Y,N11,M22,NN,CTMNN,
     1              ERRFLG)
      DO 280 J=1,M22
        DO 290 I=1,N11
          REVA(I,N11+J)=CTMNN(I,J)
  290   CONTINUE
        DO 300 I=1,M22
          REVA(I+N11,J+N11)=QQ(I,J)
  300   CONTINUE
!        EIGVA(J+N11,J+N11)=JO(1,J)
!        IF (J+1+N11.LT.NN) EIGVA(J+N11,J+1+N11)=JO(2,J)
  280 CONTINUE
C     ****************************************************************
C     *                    -1    -T                                  *
C     *  CALCULATE WTILD=P  *W*P                                     *
C     ****************************************************************
      DO 380 I=1,N
        DO 380 J=1,N
          WC(J,I)=W(J,I)
  380     CTMNN(J,I)=REVA(J,I)
      CALL CGECO(CTMNN,N,NN,IPVT,RCOND,Z)
      IF (RCOND+1.0 .EQ. 1.0) THEN ERRFLG=.TRUE.
      DO 390 J=1,N
        CALL CGESL(CTMNN,N,NN,IPVT,WC(1,J),0)
  390 CONTINUE
      CALL CTRNPOS(WC,N,N,CTMNN1)
      DO 400 J=1,N
        CALL CGESL(CTMNN,N,NN,IPVT,CTMNN1(1,J),0)
  400 CONTINUE
      CALL CTRNPOS(CTMNN1,N,N,WTILD)

C     ****************************************************************
C     *  CALCULTE (Jd)i,j                                            *
C     ****************************************************************
      NZ = 14
      DO 402 I=1,NZ
  402   NEVA(I)=1
      NEVA(14)=2
      NEVA(13)=2
      NSUMI=0
      DO 404 II=1,NZ
        NSUMJ=0
        DO 406 JJ=1,NZ
          DO 408 I=1,NEVA(II)
            DO 408 J=1,NEVA(JJ)
              ALPHA=EIGVA(NSUMI+I)+EIGVA(NSUMJ+J)
              DO 409 L=J,NEVA(JJ)
                DO 409 K=I,NEVA(II)
                 Q=K+L-I-J
                 WIJ=WTILD(NSUMI+K,NSUMJ+L)/(FACT(K-I)*FACT(L-J))
                   DO 410 P=0,Q
  410                JD(NSUMI+K,NSUMJ+L)=
     1                JD(NSUMI+K,NSUMJ+L)+
     2                (-1)**P*FACT(Q)/ALPHA**(P+1)*T**(Q-P)/FACT(Q-P)
                   JD(NSUMI+k,NSUMJ+L)=
     1                (JD(NSUMI+K,NSUMJ+L)*CEXP(ALPHA*T)-
     2                (-1)**Q*FACT(Q)/ALPHA**(Q+1))*WIJ
```

```
  409          CONTINUE
!               print *,ii,jj,nsumi, nsumj,nsumi+I,nsumj+J
  408       CONTINUE
            NSUMJ=NSUMJ+NEVA(JJ)
  406     CONTINUE
          NSUMI=NSUMI+NEVA(II)
  404 CONTINUE


C     ****************************************************************
C     *  CALCULATE INTEGRAL                                          *
C     ****************************************************************
      CALL CTRNPOS(REVA,N,N,CTMNN1)
      CALL CMATMULD(N,N,NN,REVA,N,N,NN,JD,N,N,NN,CTMNN,ERRFLG)
      CALL CMATMULD(N,N,NN,CTMNN,N,N,NN,CTMNN1,N,N,NN,JD,ERRFLG)
      SUM=0.
      DO 420 I=1,NN
        DO 420 J=1,NN
          RES(I,J)=REAL(JD(I,J))
          SUM=REAL(JAY*JD(I,J))+SUM
  420 CONTINUE
      SUM=SUM/(NN*NN)
      CALL TRACE(RES,NN,RP)      !PROV

 6000  RETURN


C        ********************* FORMAT STATEMENTS *********************
 1007 FORMAT(1X,E11.4,2X,1Hi,E10.4,2X,10F7.3,/,27X,9F7.3)
 1014 FORMAT(27X,10F7.3,/,27X,9F7.3,/)

       END


       SUBROUTINE TRACE(A,NN,RES)
       REAL A(NN,NN)
       RES=0.
       DO I=1,NN
         RES=RES+A(I,I)
       ENDDO
       RETURN
       END

         SUBROUTINE MATMULD
     G                   (NROWSA , NCOLSA ,NNA, A , NROWSB , NCOLSB,
     G                    NNB,B ,
     Y                    NROWSC , NCOLSC , NNC, C , ERRFLG )
C        ****************************************************************
       PARAMETER(NN=16)
C        ********************* ARGUMENT VARIABLES *********************
       INTEGER NROWSA , NCOLSA , NROWSB , NCOLSB , NROWSC , NCOLSC
       REAL A(NNA,NN) , B(NNB,NN) , C(NNC,NN)
       LOGICAL ERRFLG

C        ********************** LOCAL VARIABLES **********************
       INTEGER I , J , K
       REAL*8 SUM
```

```
C
C        ******************** START EXECUTABLE CODE ******************
C
C        ****************************************************************
C        *       CHECK TO MAKE SURE THAT NCOLSA = NROWSB. IF NOT        *
C        *                        THEN ERROR ABORT                      *
C        ****************************************************************
      IF( NCOLSA .NE. NROWSB )GOTO 2000

C        ****************************************************************
C        *         CHECK TO MAKE SURE THAT NROWSC = NROWSA AND          *
C        *         NCOLSC = NCOLSB ,  IF NOT THEN ERROR ABORT           *
C        ****************************************************************
      IF( NROWSC.NE.NROWSA .OR. NCOLSC.NE.NCOLSB )GOTO 3000

C        ****************************************************************
C        *                        CALCULATE  A*B=C                      *
C        ****************************************************************
      DO 1200 I = 1, NROWSA
        DO 1100 J = 1 , NCOLSB
             SUM = 0.0
             DO 1000 K = 1 , NCOLSA
                  SUM = SUM + (A(I,K)*B(K,J))
 1000        CONTINUE
             C(I,J) = SUM
 1100      CONTINUE
 1200 CONTINUE

      RETURN

C        ****************************************************************
C        *                    CONTROLLED ERROR ABORT                    *
C        ****************************************************************
 2000 WRITE(1,*)'   MATMUL,  COLUMNS OF A <> ROWS OF B :'
      ERRFLG = .TRUE.
      RETURN

 3000 WRITE(1,*)'   MATMUL,  DIMENSIONS OF C ARE INCORRECT :'
      ERRFLG = .TRUE.
      RETURN


      END


      SUBROUTINE CMATMULD
     G                   (NROWSA , NCOLSA , NNA, A , NROWSB , NCOLSB ,
     G                    NNB, B ,
     Y                    NROWSC , NCOLSC ,NNC, C , ERRFLG )

      PARAMETER(NN=16)
C        ********************* ARGUMENT VARIABLES ********************
      INTEGER NROWSA , NCOLSA , NROWSB , NCOLSB , NROWSC , NCOLSC
      COMPLEX A(nnA,nn) , B(nnB,nn) , C(nNC,nn)
      LOGICAL ERRFLG

C        ********************** LOCAL VARIABLES **********************
      INTEGER I , J , K
      COMPLEX*8 SUM
```

```
C
C        ******************** START EXECUTABLE CODE ************** ***
C
C        ********************************************************** ***
C        *      CHECK TO MAKE SURE THAT NCOLSA = NROWSB. IF NOT      *
C        *                       THEN ERROR ABORT                    *
C        ********************************************************** ***
      IF( NCOLSA .NE. NROWSB )GOTO 2000

C        ************************************************************
C        *        CHECK TO MAKE SURE THAT NROWSC = NROWSA AND        *
C        *        NCOLSC = NCOLSB ,  IF NOT THEN ERROR ABORT         *
C        ************************************************************
      IF( NROWSC.NE.NROWSA .OR. NCOLSC.NE.NCOLSB )GOTO 3000

C        ************************************************************
C        *                        CALCULATE  A*B=C                    *
C        ************************************************************
      DO 1200 I = 1, NROWSA
        DO 1100 J = 1 , NCOLSB
             SUM = 0.0
             DO 1000 K = 1 , NCOLSA
                  SUM = SUM + (A(I,K)*B(K,J))
 1000        CONTINUE
             C(I,J) = SUM
 1100      CONTINUE
 1200 CONTINUE

      RETURN

C        ************************************************************
C        *                    CONTROLLED ERROR ABORT                  *
C        ************************************************************
 2000 WRITE(1,*)'   MATMULD,  COLUMNS OF A <> ROWS OF B :'
      ERRFLG = .TRUE.
      RETURN

 3000 WRITE(1,*)'   MATMULD,  DIMENSIONS OF C ARE INCORRECT :'
      ERRFLG = .TRUE.
      RETURN


      END


      SUBROUTINE CTRNPOS
     G                   ( A  ,  N1  ,  N2 ,
     Y                     AT )
C
      PARAMETER(NN=16)
      INTEGER  N1 , N2
      COMPLEX  A(NN,NN ) , AT( NN , NN )
C
      INTEGER  I , J
C
C        ******************** START EXECUTABLE CODE ***************
C
```

```
C
C       ****************************************************************
C       *            CALCULATE THE TRANSPOSE OF MATRIX A               *
C       ****************************************************************
C
        DO 1100 I = 1 , N1
            DO 1000 J = 1 , N2
               AT( J , I ) = CONJG(A( I , J ))
 1000       CONTINUE
 1100   CONTINUE
C
C
        RETURN
        END

      FUNCTION FACT(N)
      INTEGER N
      FACT=1.
      DO 100 I=1,N
        FACT=FACT*I
  100 CONTINUE
      RETURN
      END
```

```
subroutine lyap(nf,nh,n,sf,sh,sol)
parameter(nn=16)
integer nf,nh,n,ier1,ier2
real sf(nf,n),sh(nh,n),sol(nn,nn)
double precision f(nn,nn),h(nn,nn),z(nn,nn),wr(nn),wi(nn)

do i=1,nn
  do j=1,nn
    f(i,j)=sf(i,j)
    h(i,j)=sh(i,j)
  enddo
enddo
call lypcnd(nf,nh,n,f,h,z,wr,wi,ier1,ier2)
do i=1,nn
  do j=1,nn
    sol(i,j)=h(i,j)
  enddo
enddo
return
end
```

VITA

JUERG MUELLER

Candidate for the Degree of

Master of Science

Thesis: HUMAN OPTIMAL CONTROL FOR AIRCRAFT FLYING VARIOUS STEADY STATE MANEUVERS

Major Field: Mechanical Engineering

Biographical:

Personal Data: Born in Neuchâtel, Switzerland, October 12, 1966, the son of Christian and Alice Müller.

Education: Graduated from the Gymnase Cantonal de Neuchâtel, Switzerland, July, 1985; Graduated from Swiss Institute of Technology, Zürich, January, 1991; completed the requirements for the Master of Science Degree at the Oklahoma State University, Stillwater, Oklahoma, May 1992.